Secret Sauce? Marketing Telecom Network APIs To Developers

Guenter I. Klas
Dec 01, 2012

The 1 minute takeaway

Telecoms network API exposure to software developers looks like a straightforward game in some cases and like an eternal struggle in others. The market opportunity is sized to be in the billions of dollars. Network operators are torn between going it alone, inspired by vertical ecosystems some Internet players were able to create and following the mantra of differentiation, or teaming up with others for the sake of reach and economies of scale. Apart from this, they are confronted with a mixed clientele called 'developers' for whom new approaches of business development and CRM need to be found. To make the sell, network operators need to offer more than raw APIs. That's what is explained as the 13 Strategic API Platform Elements. Moreover, the pros and cons of differentiation with network APIs are discussed, concluding that for some players API differentiation may work for particular reasons and for others not. Once a working recipe for selling and marketing network APIs has been found, the commercial opportunity to tap in appears considerable. Two case study objects are investigated: AT&T's API exposure and BlueVia.

Introduction

Several communication service providers (CSP) and mobile network operators are beefing up their capabilities to expose *application programming interfaces* (APIs) to their network enablers like messaging, location, payment/billing etc. Not that exposing network capability like SMS or location would be anything new. Interesting is that Internet companies have successfully leveraged APIs, much more than network operators have been able to do as far as API usage is concerned. This has to do with what the APIs are good for, and how they are exposed. That is different from years ago.

The big questions are: What makes CSPs successful in monetizing their network assets through telecoms APIs? What might make them more successful in the future?

A starting point is to look at the marketing activities. CSP activities achieve public visibility through the CSPs' touch points with software companies. The latter ones are considered potential business partners and business customers. The story is simple: Companies who develop applications and services for smartphones and tablets can enrich their products through tapping into the telecoms network APIs of CSPs. This drives revenues first of all for those companies and secondly for the CSPs. It also makes the apps for consumers richer, more useful and more convenient. A series of touch points with developers exists within the CSPs' *developer programs*. They are created for business development purposes; to reach out to potential business customers, to educate customers about commercial opportunities and technology alike, and to get them to try the sweets in the shop (speak test the telecoms APIs and eventually adopt them in services and applications). Thus, looking more closely at such developer programs is revealing and some results are shown below. Furthermore, thoughts are added related to market opportunities and strategy.

What's happening out there?

Most key players in the mobile industry have been working on network enabler exposure via APIs for some time (from US firms like AT&T, Verizon Wireless to European companies like Telefonica, Deutsche Telekom, Orange/FT, Vodafone and Asian CSPs). The mobile industry has created standard definitions of APIs through the GSMA, published under the name OneAPI [1]. Though a successful Canadian trial [2] was conducted amongst Canadian network operators who implemented the OneAPI

interface specifications, wider harmonised cross-operator adoption is still pending and in the works e.g. in the GSMA.

The Dilemma

CSPs are wondering about how to approach this opportunity. Being competitors, they are driven towards differentiation. In the extreme, differentiation can mean deployment and marketing of completely different application programming interfaces. The GSMA [3] has nearly 800 operator (carrier) members. If every member went for their own proprietary APIs and tried to market them to software companies, the end would be near: It would be called lack of critical mass and total fragmentation. But this worst case is also unlikely to occur, because most CSPs wouldn't be able to afford such a strategy: The required efforts and costs are too high.

That fear drives them towards standardisation of APIs, however standardisation can be slow, and result in agreement on the lowest common denominator which might not always reflect the latest, newest innovation ideas. Unfortunately, cross-operator common API exposure across geographies doesn't only require agreement on syntax and semantic of programming interfaces. It also requires some supporting infrastructure. That needs to be defined, implemented, deployed and operated. And CSPs would need to get their own platforms ready to interwork with such common support infrastructure. That's complex and requires investment.

Not surprisingly, reality lies somewhere in the middle. Standardisation of syntax and semantic of APIs has been significantly progressed through the GSMA's OneAPI initiative [4], whilst several CSPs set out to explore and innovate on their own, in a quest to discover the key which unlocks the riches promised by APIs. Two initiatives are illustrative: AT&T's API exposure program [5] in the US and the BlueVia initiative [6] started by Telefonica in Europe and now supported by Telenor [30].

What's the issue?

If someone had identified the value-adding APIs which enable new value-adding applications and services, had exposed them to meet the needs of a significant developer community, had wrapped them into a diverse set of business models and offered them at the right price to generate win-win, then that someone would likely ride a wave of success and have a chance to recoup required investment or even make a profit.

There has been some good news in 2012, but not overwhelmingly much, indicating that the challenge is non-trivial. Which approaches have the aura of a degree of success? Which ingredients have they been trying out? Can we throw those into a pot and cook them up to distil a secret sauce of how to successfully design and market carrier network APIs?

For now, the answer is pending. Let's start the search by examining the two aforementioned API initiatives. However, before we do that, let's consider a potential ideal reference scenario.

The reference scenario

This shall be a hypothetical telecoms service provider. They have various telecoms core assets which they strive to monetise. This includes text messaging (SMS), multimedia messaging (MMS), geo-location of subscribers, context of subscribers (the speed of the network their phone is on, the capabilities of their smartphone,…), payment and billing infrastructure, and information related to subscriber identity.

They want to enable innovative smartphone/tablet and desktop applications which make the life of people easier, more enjoyable, more efficient and effective. They want to enable an application of FoodIsMyPleasure (FismP, pronounce 'fismip'), a hypothetical London Internet start-up.

FismP have cooked up the following Android app: It helps you search for bars and restaurants, shows your favourite place on a map and where your phone is at the moment. It can indicate whether any of your Facebook or LinkedIn friends are in the area. For a micro-payment it allows you to download any

recommended consumer-generated stories (text, pictures, audio, or video) linked to this place (someone fell out with someone else, tripped up a waiter, met a celebrity or fell in love with someone in the bar…). Once you have opted in to some T&Cs, you can type your own mini-story and upload it to the cloud as an author-on-the-go.

FismP's cloud-based analytics engine, powered by the carrier's Platform as a Service infrastructure wades through the user stories, establishes affinities and minimum distances between authors in a 10-dimensional space of parameters expressing user preferences, habits, likes and dislikes. For a £0.99 micro-payment you can purchase the result and get in touch with the identified person if you like and they have opted in. You can recommend the downloaded story to others. Depending on what you revealed about yourself the app notifies you about special sales offers along the route to your restaurant once you move within a radius of 200 meters of any relevant shops.

This app uses several of the carrier APIs: in-application payment API, SMS for billing notification, a geo-fencing API, a context API to discover whether your Android phone is on GPRS, Edge, 3G or even superfast LTE (thus, the app can decide to ship the downloaded story as text only or including audio/video). It also makes use of a Google map API, a Facebook and LinkedIn API.

You like this app. It is fun, it's sticky. 1.2 million of similarly wired people like it too. The carrier is able to throttle traffic on their APIs to avoid breakdown of their backend IT systems. They do API analytics (which APIs are used most; what app features are the big hits; is the API response fast enough …).

The carrier has pointed FismP to sources which explain how to program the application such that it doesn't empty the battery of your Android phone within an hour. Also the carrier people explained to the app people what to watch out for such that the app behaves network-friendly and doesn't participate in creating signalling storms when 1.2 million restaurant goers try it out simultaneously (The storm could be worse if FismP's server were down, however this is unlikely as FismP's server-side software runs in the carrier's Cloud). The FismP team has learnt about the carrier's APIs at one of the carrier's developer conferences. They followed up at a local hackathon to try out the APIs and things looked promising. To check out how an early app prototype would work on an Edge network and a 3G network, FismP got access to the carrier's network infrastructure and the carrier's network and terminal experts (actually they got some temporary office space at the carrier's labs). During the app design phase, the carrier pointed them to the company which later built their analytics engine. The chief nerd of that analytics company happened to sometimes hang out in the same office space.

End of last week, FismP had 1.2 million active app users. This week, FismP is launching their app for use in another 5 countries. FismP is setting price points for micro-payments in 3 different currencies, this with a few mouse clicks on the carrier's developer web portal. Revenue is flowing in multiple ways: The revenue from app downloads from Google is shared with FismP. In-application advertising revenues flow mostly to FismP, so does the revenue from micro-payments. Everybody gets a cut.

The FismP team told the carrier that the payment API was unwieldy and overly complex to implement. Promptly the carrier published a simpler one within 6 months. FismP is a bunch of software developers, 16 in total. The carrier has 160,000 employees on its payroll worldwide. Still, FismP feels like a valued partner, not a tiny fish hitting its nose against a colossus of a wale. The news spread. This week, another 5 FismPs knock at the carrier's door. With a key difference: Each of them comes with a new idea of application and wants the carrier to help them deploy it across iPhone, Galaxy, Lumia and BlackBerry, at minimal cost using HTML5. And by the way, it should work in the US, Europe, Latin America and Asia.

Now then, let's step into reality…

Focusing on two: AT&T and BlueVia initiative

Both AT&T and Telefonica have done a good job in 2012 in pushing the boundaries of telecoms API exposure and getting closer to the reference scenario depicted above.

AT&T is promoting telecoms enabler APIs through their AT&T developer program [5]. Telefonica launched the BlueVia initiative [6] to expose APIs and has recently managed to win Telenor to join them for BlueVia.

Looking at AT&T's approach and the BlueVia way of life, I noticed several common aspects, but equally some striking differences. Both seem to enjoy a degree of success, though it's not fully clear how developer outreach and media attention correlate with revenue and profit generation.

What are the strategic objectives?

Obviously various ones are possible and each organisation will craft their own ones. There is the 'role model' API exposure done by non-telco companies like Amazon, Facebook, Netflix and many others. Exposing APIs is a way of doing business, from payment services, product search, to travel booking and entertainment.

So from a CSP point of view, why not utilise network assets like geo-location information, billing engines, databases with customer/subscriber and wireless devices information more effectively and monetize them through new channels? Thus, an obvious objective will be to monetize those assets through providing "a-lets-do-business" interface to interested companies. Those interfaces offered in good 2010+ style are Web/Internet APIs. Both our case study examples AT&T and BlueVia will most likely have that objective.

A second, maybe less visible goal, is to make use of CSP network assets more effectively and reduce the amount of CAPEX required otherwise.

The first objective of monetizing network assets through APIs has been discussed in various places, less so from a CSP perspective than from a developer perspective (as in Developer Economics 2012 [7]).

The second objective of curbing unnecessary growth in telco network-related CAPEX might be even more challenging to implement through CSPs' API exposure. How would channelling B2B business through carefully defined APIs lower network CAPEX? An example would be a Context API [8], which allows e.g. a news/media service to discover the wireless transport bearer a smartphone/tablet is currently using. Is it GPRS, 3G or LTE? The media content provider can then adjust the media types and amount of data sent to the smartphone with two effects: (i) It improves the customer experience (better response time) and (ii) it doesn't overload the CSP's network. It simply leads to a smarter, more intelligent use of deployed network infrastructure.

Who is the customer?

In essence: Who needs telecoms APIs? The answer of the industry has been for a while: Big businesses and geeks (the sandal-equipped garage developers). They stand for two major categories also denoted short tail and long tail. The short tail is sometimes called the short head. The short tail stands for a few big companies offering big revenue potential. The long tail stands for a huge number of small and smaller companies down to the size of individuals whom the CSPs also regard as potential B2B customers.

Engagement with the short tail is well understood. Bilateral contracts e.g. between CSPs and say Google, Nokia, RIM, etc. for carrier billing may be one example. You go to the Google Play shop, buy some digital good and at checkout you get it charged to your carrier's bill.

In contrast, engagement with the long tail is much less understood. The long tail has been overhyped for some time. It's a promising customer group: Lots of potential B2B customers, all with smaller pockets (at least when they start out) but high entrepreneurial spirit and innovation potential. The next Netflixes or Twilios or Rovios (3 students + a business angel + a game like Angry Birds). Doing business with the whole of the long tail is more an art than a science. I suggested in the past that the

long tail doesn't even exist because it is not organised. Whom to speak to? It neither has an industry association that would lobby for it nor a union that would spell trouble for it. It's jelly. Standing out from the crowd in their attempt to nail this jelly to the wall has been BlueVia [9].

Identifying the 'customer' for telecom network APIs is crucial. Depending on 'who' the customer is, customer engagement, sales force and customer relationship management (CRM) of the CSP will look different.

Extreme case 1: The B2B customer is a firm like Nokia or Google. Most likely, customer engagement by the CSP is highly targeted, customised, and bilateral, with toing and froing involved and legal departments of both sides raising their concerned voices and eventually giving their sign-off.

Extreme case 2: The B2B customer is a very small start-up, a bunch of mostly software developers. These guys don't have an Account Manager for CSP xyz nor do they have corporate lawyers. They have ideas. They are comfortable in being addressed in different ways: discovered at developer outreach events performed by the CSP; invited to show off their ideas and preference for Apple Macs; happy to discuss with likeminded people from the telco organisation wearing the same brand of jeans as they do. Apart from computers, monitors, a selection of smartphones and tablets, their most valuable asset is their coffee machine.

Within this spectrum of developers Vision Mobile [7] has identified eight developer segments (Hobbyists, the Explorers, the Hunters, the Guns for Hire, the Product Extenders, the Digital Media Publishers, the Gold Seekers and the Corporate IT). I recommend having a look at this unconventional customer segmentation. In my view, CSPs largely miss groups which lie within the boundaries of above extreme cases. Stay in the middle of the road and you get run over (and looked over). In the middle we find e.g. ISVs, mobile marketing agencies, system integrators and outsourced software development specialists. These are influencers and doers who should know about a new telecom API story. If there is truth in this kind of market segmentation, it further implies that the CRM of network operators focussing on 'developers' could in many cases do with a significant overhaul (if it exists at all and is affordable).

Are apps development companies exclusively waiting for CSPs to provide them with good offers and convincing USPs? Unfortunately not. The same developer companies are lured into the gardens of Apple, Google, Facebook, Samsung, Amazon, Twilio, Netflix, Appcelerator, Sencha, Bango, Boku etc. All of them offer APIs, supporting tools and other candy. This has mighty consequences for most of the CSPs. The CSP's API offering becomes easier to compare to other offers in the market. Developers will quickly find out: What is good fashion (e.g. RESTful API), what looks outdated and what's overpriced. A CSP who is unable to benchmark their API exposure effort to the ones of other leaders in the field is like Marks & Spencer in the UK trying to sell clothes to a 25 year-old enthusiast of en vogue fashion retailers Next or New Look.

Moreover, many CSPs are at risk of misjudging the market. To quote from [7], "We have all come across the perennial cliché of the developer as an unshaven geek with long hair and sandals. This image is outdated." CSPs would be well advised to go and get to know their 'customers', this time also 'developers' = application software developing companies, not just end consumers.

Where to reach the customer?

Network operators have a good history of doing business with enterprises. There is also a good history of selling APIs to a few, select top notch heavy-weight partners (e.g. for carrier billing). When it comes to selling telecoms APIs to apps developers, well, shall the API proposition be sold to their CEO in Europe? Or to the CTO in the USA? Or to the lead engineer and the software developers working for that company? Today there is a relevant trend to outsource apps development, e.g. to a team of software engineers in India. So then, a look behind the apps industry structure might be useful to figure out where to do physical developer outreach and in which language. What's the logic then of

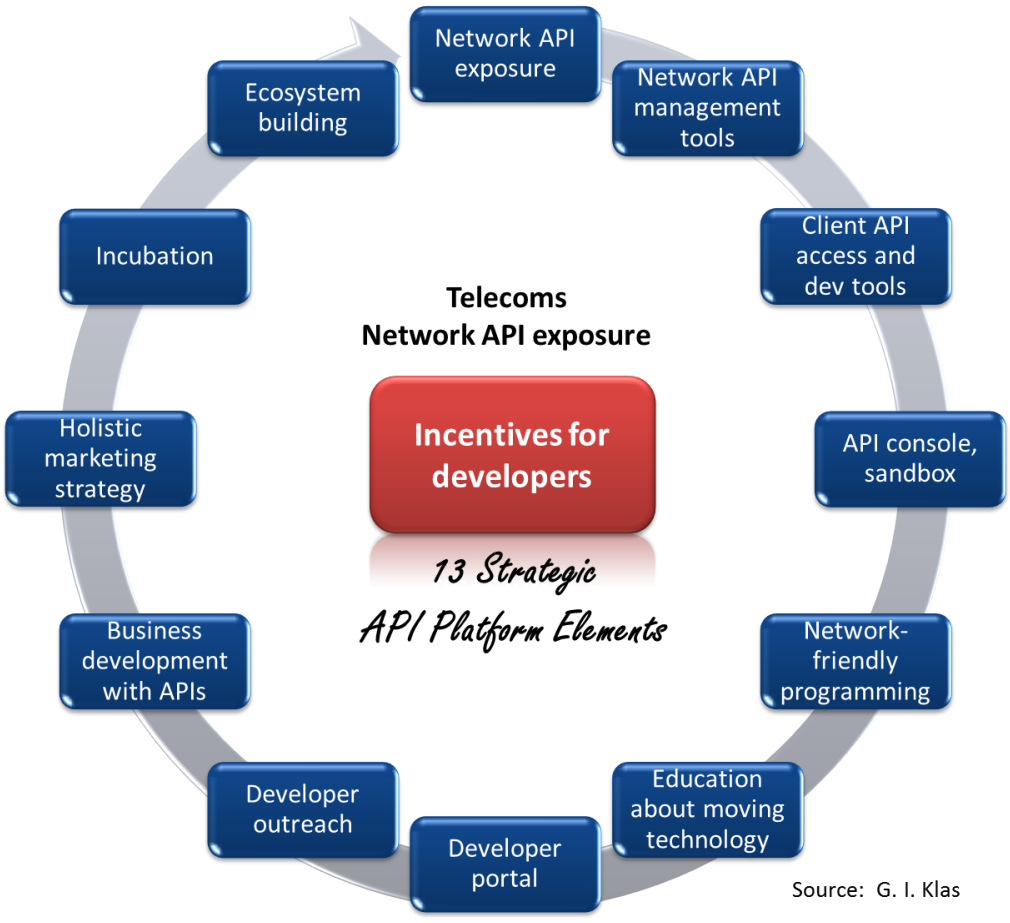
running hackathons in the future in London and New York when the developers and software architects of the next millions of apps reside in emerging markets?

What's the product to sell?

From a telco perspective several products or services can be sold particularly to software companies and developers through APIs. Examples are: In-application payment and billing services, geo-fencing services, smartphone and user context information etc. But there is more collateral to those API-related sales: The CSP can equally sell other services to the same developer companies, like access to mobile advertising, to yellow pages business information, cloud storage, cloud backup etc.

What's required to execute the strategy? Strategic API Platform Elements

A number of key ingredients are shown in the picture below.



Source: G. I. Klas

1) Network API Exposure

First, CSPs need to have working APIs in place. Second, the APIs need to be offered in a simple, modern, and flexible way. To give examples: AT&T offer a series of APIs [10], at least SMS, MMS, WAP Push, Location, Device Capabilities, Payment, Notary Management. In a similar way does BlueVia [11].

Important is to offer the APIs in the style demanded these days. There is a bit of fashion sense even with software developers. Where big corporates have still enough money and experts to deal with technologies like SOAP and WSDL, smaller and nimbler organisations favour RESTful APIs based on HTTP. Those technologies refer to the way how information is queried from the telco's API gateway.

Equally important is the simplicity in telling developers how these APIs actually work. This happens in the form of API specifications, most often in the form of a few lines on a website (e.g. the BlueVia payment API [12] or AT&T Device Capabilities API [13]) and stories about how such APIs are used.

Equally important is the provision of sample applications and sample code. So I go to AT&T's developer site, think I feel like programming in C# today, choose C# as my language, click on Device Capabilities and I'm shown --- well then, I better choose RESTful PHP, and indeed it shows me some PHP script code. To get to some application samples with BlueVia, I'm directed to Github.

Apart from this we of course need the entire infrastructure sitting behind an API: some API gateway, working interfaces to IT systems from network enablers to settlement and reporting systems.

2) Network API Management Tools

Such tools relate to

- traffic management of APIs (e.g. limiting access and over-usage through transaction throttling or setting quotas for API use by developers),
- functions that handle authorization and authentication,
- software that protects the API gateway from unauthorized access and denial-of-service attacks,
- features that ensure data protection,
- software that performs API analytics to understand the usage of APIs and
- software for API user management.

Such professional API wrap is particularly useful, once the API traffic breaks significantly above the first barrier of say a mere 100 API calls per day, more in the direction of Twitter's 13 billion API calls per day from May 2011. In the latter case, security, scalability, reliability and smooth operation are important.

Writing up a telco API specification (a few lines of script code put online), implementing the API in a gateway and issuing a press release is called exposure of 'raw APIs'. For the API company Apigee, such APIs are naked [14]. Time has moved on. People are wearing fine cloth today.

3) Client network API access and developer tools

If a game developer wants to offer their end consumers the feature to buy additional game levels, an in-application payment API is the right thing. The end consumer is presented with a simple payment page: Confirm purchase of next game level: price = $1.99, will be charged to your carrier's bill. Click CONFIRM, T&Cs apply. Now, this payment API has an endpoint on the telco side and needs an access point in the smartphone application itself. The latter one needs to be created by the game developer. For this, the software people need tools to drive cost efficiency, when they plan to engineer such API endpoints: for the Java version of an application on an Android phone, for the HTML5 version of the same app so it can run on iPhone, the forthcoming Firefox OS smartphones and some Samsung tablets, for the slick Objective-C-based version of the app running on the iOS operating system of the next iPhone. Good luck without tools.

CSPs provide tools to different extents. For example, AT&T points to a series of software development kits [15] (e.g. for Android, Windows Phone, BlackBerry etc.). They also recommend a toolkit for HTML5-based API access for developers who prefer apps development using HTML5, JavaScript and CSS. Equally AT&T has pointed developers to the use of client-side frameworks like Sencha Touch that speed up client-side development and serve as cross-platform tools too. Sencha Touch for instance assists developers in their application design with themes, off-the-shelf code for forms, charts, handling touch events, lists, carousel items and the like. Similarly, BlueVia offer a set of code libraries [16] that serve the purpose of wrapping the BlueVia APIs for use with Android, .NET, PHP, Ruby; this under the column 'Tech Stuff'.

CSPs will need to take care of hygiene factors in this API business. Just declaring an API is ready to be used won't do the trick. The competition in the form of over-the-top players is not sleeping, instead relentlessly suggesting they understand the needs of innovative new businesses better and are keen to offer them simplicity and short ramp-up time.

4) API Console, sandbox

This is often an on-line try-it-out service for developers to play around with the APIs of the CSP in order to understand how the different APIs work, what syntax and semantic the APIs impose on developers and what exactly is shipped from say an application on Android to the telco's API gateway in terms of messages. An example is the AT&T API Console [17]. A sandboxed environment to experiment with the offered network APIs goes in a similar direction.

5) Education about network-friendly programming

It is in the interest of CSPs to tell their B2B customers how to program applications such that they avoid two things: (i) devices with empty batteries in the hands of disgruntled consumers, (ii) networks overwhelmed by a flood of unnecessary signalling messages caused by apps and their corresponding application servers. Checking this for our two case studies, we find that AT&T promote the network friendly design of applications through multiple ways, including links to best practices for how to program decent apps [18] and to the AT&T ARO tool (Application Resource Optimizer) that enables developers to measure how well or bad their application performs on a mobile network [19].

ARO uses a data collector on the smartphone side to record various pieces of information. Then the trace file is shipped to analysis software which may run on a laptop. The ARO analyser is downloadable from AT&T and made freely available.

At the Developer Summit 2012, AT&T explained to developers in a rather unique way the rationale for network-friendly apps, and gave advice on how to develop apps in the best way.

6) Education about the moving technology base

What happens to an eco-system when parts of its foundation age, or when disruptive technology emerges? The whole system may fall apart (as one may see with Symbian). Thus, educating B2B customers about what the near future may bring firstly helps to safeguard a young ecosystem against unanticipated threats and secondly highlights potential commercial opportunities which wait just round the corner and aren't that visible today. For instance, AT&T explain to apps software developers what the future brings in terms of new cellular radio technology like LTE and which new steps are undertaken by chipset vendors like ST-Ericsson, Texas Instruments, Broadcom, NVIDIA and the like. One may argue chipsets are far away from software development in HTML5 or Java, however innovation on chipset level (e.g. optimisations for graphics in HTML5, WebGL and others) is yet another enabler for the overall apps and API ecosystem.

7) Developer portal

This is a website that is the front-end of the CSP's B2B customer relationship management. A top notch developer portal provides amongst others education about the CSP's APIs, offers API-related tools, developer registration (including bank details for settlement purposes), references to deeper ways of collaboration with the CSP (e.g. through their venture capital arm or their collaboration zone), and most important an online process for getting a new application tested, certified and ready for launch in any preferred app store, market place or play garden.

Through the developer portal, developers may also be offered analytics for API usage. Through this, developers and CSPs can rank APIs in terms of popularity, throughput and API features which are in demand or out of favour.

The style of developer portals tells a lot: (i) about the target audience, (ii) how the API exposure team wants to be seen in the eyes of their corporate customers, and (iii) how holistic the outreach to developers is planned to be. In this regard, it's illustrative to compare BlueVia and AT&T.

8) Developer outreach

Other possible names would be CRM (customer relationship management), B2B Client Management, API Marketing & Sales…. Naming is indeed not to be discounted. Decision makers in a CSP with a telco-biased culture may not have any affinity with the notion of 'developer', mistakenly believing this mysterious 'developer' is largely irrelevant for their billion dollar business. Another CSP might want to 'modernize' their culture and wrap themselves in the cloths of an Internet company like Google, Yahoo, or Facebook, and thus adopt speak and slang common in those circles. Then 'developer' has a very important co-notation. It stands for software entrepreneurs, creative brains, founders of start-ups, clever people pregnant with the next breakthrough Internet ideas.

The business development approaches vary widely across CSPs when it comes to detail, though several use common ingredients. Those include major developer conferences (like AT&T's Developer Summit, the last one in Jan 2012, the next one in Jan 2013 at the Palms Casino Resort in Las Vegas, or BlueVia's many sponsored events as per their website). A further must-have ingredient is the hackathon.

A hackathon is a developer outreach event, designed more often for technical than non-technical attendees. During a hackathon, participants build mobile applications in real-time, in a relaxed, and often competitive atmosphere with some prizes waiting for the winners. Hackathons can be a double-edged sword: If arranged well and staffed by the CSP with top technical experts, they generate impact, improve brand awareness and drive credibility. If executed poorly hackathons can backfire, e.g. when they are peppered with serious technical glitches or if CSP executives create culture clashes when lecturing in suits to software engineers hanging out in T-shirts and jeans on bean bags with laptops on their laps.

Often, developer outreach activities are organised into a Developer Program which is marketed to developers. Such a program will arrange for those developer conferences and hackathons. It will further support software developers through CSP expert blogs, forums, webcasts, webinars and face to face training events.

9) Business development with APIs

This refers to efforts to explain to customers (= developers) what the art of the possible is when customers have access to telecom APIs. Assume a CSP offers a geo-fencing RESTful API. So then, what to do with it? The proactive developer outreach of AT&T (at their 2012 Developer Summit [20]) explains how this API can be used to implement location-based marketing in shopping apps, how to combine this with an SMS API, and how to build on this for location-based marketing analytics. This is about planting ideas and spreading creative use cases. BlueVia has a section called inspiration [21], where BlueVia API customers share what they have created with the help of APIs. Again, the idea is to go beyond the naked API exposure.

10) Holistic marketing strategy

It is one thing for a CSP to promote their RESTful web APIs say for SMS (text messaging) or identity (Hey developers, go to web link xyz and there you find our new sexy API!). It's another thing to carefully tie things together and sell more than just the API access but equally additional, complementary, value-added services.

This includes promoting telco APIs in the context of different sectors like automotive or the machine-to-machine industry.

Examples include AT&Ts knitting together of APIs with their IP TV business, the promotion of complementary cloud services like Storage as a Service and Compute as a Service plus AT&T PaaS offerings [20].

To pick an example, AT&T promotes the creation of U-verse enabled apps for app interaction with digital TV (e.g. TV as display, app as remote control + TV guide). A device associates with a digital TV

receiver, then can use U-verse Enabled APIs to send commands to and retrieve information from the receiver. Such an API may also provide access to complementary TV channel data.

The cross-marketing strategy may also include promoting APIs of other CSP divisions through the same developer outreach channel. Back to our case study, AT&T promotes an API from another AT&T division, the Yellow Pages business.

Some CSPs offer apps developers not only help in creating their apps and enriching them with telco APIs, but also offer the apps a direct channel into the CSP's app store. An example of this has been AT&T, offering support to get a developer's application straight into their AT&T AppCenter, where the claimed USPs compared to Google Play and other app market places typically are: better apps discoverability and quality-assured content. Despite that, some statistics from June 2012 tell that the use of telco app stores as distribution channels has dropped to 3% of developers [7].

Overall, AT&T's developer outreach appears particularly holistic. The areas they cover in developer outreach range from network-friendly apps design, apps impact on the CSP's network, awareness of terminal and network technology roadmaps, integration with mobile advertising to design for security and dealing with privacy issues.

11) Incubation

Several CSPs have realised that a gentle nurturing is often useful to get the ball rolling and to lower the hurdles for potential B2B customers to engage with telecom services and APIs. Both of our two case study objects have put in place related incubation zones. AT&T operate their AT&T Foundry [22], Telefonica operate their incubator called Wayra [23].

The AT&T Foundry for example offers interested companies workspace, the opportunity to work on vertical sector projects (e.g. for mHealth) and project coaches. The Foundry is also a meeting place for management and experts from AT&T and those interested companies. Goal is to work together and collaborate on new products and services. Speed dating and technology demonstrations are a typical ingredient of an incubator. An objective is short time to market and fast project cycle time.

Equally the CSP may offer a company access to the CSPs network for advanced testing purposes. Some CSP incubators also knit venture capital support into the overall proposition. An example for that is Telefonica's Wayra, which is also promoted to developers through BlueVia.

Some CSP incubators keep the relationship between CSP and 3rd party company a 1:1 relationship. Others promote cross-fertilisation between the companies under the same incubator umbrella. The AT&T Foundry is an example of that if we believe one of their claimed success stories.

As one would expect, incubators tend to be located in areas which have proven to be breeding places of innovation. Thus, we find AT&T Foundry branches e.g. in California, Texas, and Israel.

12) Ecosystem building

The logic behind this is the recognition that lots of collateral is required to get a core business idea to successfully take off. In many cases, this relates to investment for which it's difficult to generate a business case for it in isolation. Many such investments contribute to the bottom line only indirectly. They cost money, but it's difficult to attribute any revenue directly to them.

As demonstrated at their major annual developer conference in 2012, AT&T incorporate vendors directly into their developer outreach story under the brand of AT&T (e.g. Samsung as smartphone innovator, Microsoft and RIM as platform providers). This requires extra convincing and coordination effort, likely with no immediate payback. It's about building an ecosystem and that requires some scaffolding and supporting pillars.

13) Incentives for developers

The importance of interesting commercial models and incentives for developers has been pointed out e.g. by Telco 2.0 Research in [27]: "Operators have not yet made it sufficiently attractive for

developers and content owners to build solutions that incorporate network APIs". I like to add that when we go shopping we tend to not simply buy stuff because it has a good price (commercially attractive). We try to understand what we can do with the stuff. Thus, the seller needs to explain the benefits and opportunities to the buyer. Ergo: CSPs need to explain the benefits of using telecom APIs inside apps. This relates much to the points made above on 'business development' and 'holistic marketing strategy". Commercial incentives are not good enough. The downstream opportunities from using telecom APIs must be explained in vivid colours to apps developers. AT&T has done a great job on the latter at their 2012 Developer Summit. BlueVia was apparently first in innovating with new commercial models and actually paying developers instead of charging them (e.g. paying 10% of revenues from sent messages, [28]).

Unique selling points of telecom operators

As Developer Economics argued, Facebook could become an increasingly attractive platform for apps developers. The platform is over the top, not bothering about smartphone and tablet operating systems. And Facebook can offer APIs similar to the ones which CSPs are offering today: e.g. an API for messaging, an API for voice and video calls (via webRTC), payment, and location via GPS.

What is the competitive edge of CSPs in this game? Well, it lies in their closeness to the networks and IT systems (like billing) which they operate, in the ability to improve quality of service and provide QoS on demand, in the data they get from and about customers and their devices, their insight into all kinds of wireless consumer, business and industrial (M2M) devices and their ability to work with any kind of vertical sector (from automotive, health to utilities and insurance), from SMEs to multi-nationals.

Is telecom network API success dependent on smartphone/tablet operating systems?

This question is relevant insofar, as telecom APIs must be accessed from inside an application directly on a smartphone or tablet or from the application server the device-resident app is connected to.

Assume, a lot of apps make a call to APIs directly in their software code on the phone (e.g. by issuing HTTP commands in good REST style). From a CSP perspective it will be interesting to recall that the dominant device OS platforms are iOS (Apple) and Android (Google). A study from June [7] states that developers are abandoning other platforms at impressive rates (BREW (Qualcomm): 60% of developers plan to jump ship, Bada (Samsung): 49% of developers, BlackBerry (RIM): 41% tend to not develop any apps for it anymore). Windows Phone is said to gain developer mind share.

What it means: There is a degree of unfortunate dependency. CSPs are dependent on getting their network API endpoints exposed in iOS, Android and Windows Phone with priority. But then, at least Apple and Google are not ranked by CSPs in the top league of the most network operator-friendly and open-minded cooperation partners. Native applications usually have good access to native platform APIs, but those are dictated largely by Apple and Google for their platforms. This can present a bottleneck for CSPs. Ideally, CSPs like simple, elegant API end points exposed through the development environments used for creating iOS and Android applications. That explains at least partly the importance of SDKs (to wrap API endpoints into the relevant programming language environment that is prevalent on the different operating systems) and the reason why CSPs like the deployment of HTML5/JavaScript apps via browsers and web runtimes. An open system like Mozilla's Firefox OS, which can be shaped by CSPs sounds like a paradise in this regard.

Differentiation versus going it alone

Should CSPs strive to harmonise the telecom APIs they offer to corporate developers? There are pros and cons to this.

Benefits of API harmonisation include

- Less effort for developers who want to create apps for consumer use across operators (e.g. a network context-aware application to be marketed across Europe for use on any mobile network like O2, T-Mobile, Vodafone etc.).
- Greater usage of network enabler infrastructure of a given network operator. E.g. if a US developer registers with AT&T's developer program and the app makes use of an AT&T device capability API, this app will work nicely on the AT&T network. If European network operators support the same API, the app can be sold to European consumers as well, who use the networks of European carriers. The app could then make direct use of the APIs exposed by the European operators. That's where more traffic to (here the European) network enablers is generated from the US application (like to a database that knows about a user's device capabilities and a user's context and preferences).
- Increased size of the addressable market for a developer company. In above example the apps download market would go beyond the US and include European countries as well.

Downsides of harmonised API implementation and exposure

- It can take a long time, requires consensus and (quasi) standardisation processes.
- It most likely requires extra investments in infrastructure to support the harmonised APIs.

In contrast to API harmonisation as suggested by the GSMA OneAPI Initiative stands the call for differentiation. Whether differentiation in telecom APIs makes sense depends on additional factors. It's also about eyeballs and attention span. Software companies have limited resources and channel them to the most profitable areas. Earlier in 2012 the Developer Economics report [7] estimated a developer mindshare of 76% for Android and 66% for Apple's iOS operating system. The report cites user reach as the top platform selection criterion used by developers. This tells a lot about what drives the interest and attention of developers.

The current situation speaks in favour of iOS and Android, magnets of developer attraction. But does it speak in favour of proprietary AT&T or BlueVia APIs? The former are only supported on AT&T's US network, the latter only by some Telefonica and Telenor operations? The urge to differentiate when done with the APIs (namely syntax and semantic, having different APIs like sweets in a candy store) can be a shot in the own leg as it works not in favour of end consumer reach, which as we learnt is of prime importance for developers. If a company develops a smartphone/tablet application that uses AT&T APIs, then the company reaches many US customers, but has to rework the software to let it be of benefit to European customers say on the Telefonica, Vodafone, or Deutsche Telekom networks. That's not exactly cost efficient.

Also, important to be clear: Companies can differentiate through API syntax/semantic, but equally through many other features (new APIs, API platform performance, developer outreach, business models etc.).

Those who argue that CSPs must differentiate *with syntax and semantic of APIs* implicitly discount the feedback of developers who say their number 1 criterion to choose a platform (with all its fancy differentiated features) is reach and number of eyeballs. It's apparently not the fancy bells and whistles of a proprietary API. However, the argument I just made can fall apart when the party who differentiates can indeed deliver (on its own or through an alliance) enormous reach and sufficient number of eyeballs.

Now assume consumers download their apps mostly in the country where they live, go to work and meet their friends. As far as telecom APIs are concerned, an app would tap into the backend of the carrier network which a consumer is using in that country. Next take the following statistics from [7]: In 2012, developers identified North America as the biggest apps download market (41%). Equally, developers have reported pay-per-download as the most important revenue source in 2012. Thus, the US market is somehow unique in terms of (i) size of global consumer app demand, and (ii) sufficiency for revenue generation for developers. This because app demand implies app downloads and since

pay-per-download is the most widespread business model, it also generates good revenue. For many developers it will today generate revenues sufficient enough to not bother about other geographical markets and to not demand cross-country/cross-operator harmonised network APIs.

How important is it then for US network operators like AT&T to harmonise their telecom API exposure with their colleagues from Spain, Germany, Russia, and India? Today, not that important one may argue. Only looking at the monetization through apps downloads, the US market is large and thus a great incentive for apps developers (there is no need to be greedy). Getting those apps fitted now with proprietary telecom APIs is possibly a greater opportunity than waiting for international harmonisation of telecom APIs plus implementation in API gateways around the globe. The latter risks letting the opportunity slip if we believe in the hypothesis that the developed markets will one day get saturated with apps (also driven by already high smartphone penetration) and the next 10 million of apps will be created for use in emerging markets as forecast in [7].

In contrast, using the same statistics, harmonising telecom APIs for download markets in Asia, Latin America and Africa would make those markets together an even bigger opportunity (48%) than the US alone (41%) in terms of apps demand. Is this the logic of Telefonica and Telenor with BlueVia? It would fit to the idea of promoting BlueVia APIs in Brazil, complemented with the launch of a web-based phone by Telefonica in H1 2013. Telenor also has some emerging market footprint (India, Pakistan, …).

To quote from [7] (fair to say that the report was co-sponsored by BlueVia): “The next 10 million apps [added: which can make use of telco APIs] are not going to come from the current leading markets, but from BRIC (Brazil-Russia-India-China) demand for localised apps.” Today, we could add Africa (back in 2001, Africa didn’t make it into Jim O’Neill’s concept of developing economies).

From this analysis one could argue: In the short-term, it’s important for AT&T to get the US apps market equipped with telecom APIs. In the mid- to long-term, the apps growth rate may decelerate in the US. For BlueVia the short-term opportunity in Europe appears smaller. The story would be more about investing in the short-term for a bright future, i.e. establishing BlueVia APIs in those regions which promise rising smartphone penetration together with growing apps literacy and within those big regions in countries where coverage with wireless broadband networks is likely to happen first: Somewhere in Latin America (e.g. Brazil), Asia (incl. India + China), Russia and Africa.

Money making machine?

To which extent then is telecoms network API exposure a component for a profit-generating engine? The answer to that question will be known by the corporate finance teams of the big players who have invested millions of dollars and euros. The question relates to more than the economics of apps for developers.

Apps can generate revenue. Software developers develop mobile applications to generate profit. The foundation of revenue is reach with consumers (many millions of eyeballs is good). On that foundation they build revenue from pay per download, freemium models (free download, payment for an upgrade or premium content), in-application payment (for an application feature), subscriptions, in-application advertising, royalties from pre-installed applications on phones and others. A game like Fruit Ninja creates $1 million per month in total revenue for the developer [24]. CSPs can make a cut from those good revenue streams. But then other reports (including a controversial story in the New York Times from Nov 2012) say that lots of apps developers don’t generate sufficient revenues to secure a profit [25], [26]. So money goes round, though not to everybody in equal measures.

CSPs are not only dependent on successful software developers using the telecom APIs. The stack of hurdles is much higher: First, apps developers must be convinced to make use of say AT&T or BlueVia APIs. Once an app developer is “signed up” the application must actually be used over time. Otherwise it would only generate e.g. a one-off revenue spike (from app download) but little in-application payment and advertising revenue. A problem is that end consumer engagement with a

given application falls sharply after download of the app (a figure from [7]: only 24% of users continue using an app three months after download). What's then happening with all the potential in-app advertising and in-app payment revenue? If consumer engagement with many apps drops off quickly after download, the more important it is for CSPs to see a continuous stream of new apps coming online *over time*, all equipped with network APIs. That speaks in favour of a CSP undertaking continuous and relentless multi-year CRM and developer outreach, in promising geographies and in local language.

Above is just one example to show that CSPs don't have full control over the cash generation ability of network APIs. However, they can influence the odds of success. Despite all the finer details and the complexity of the overall endeavour, one thing should become clear: Stuff must be attractive. Not only the apps created by developers but also the shopping malls and candy stores expected by corporate developers from network operators (see the 13 points earlier). This gets us closer to something like the reference scenario described above.

Overall, according to another report [29], the future is bright for telecommunications network APIs: Quote: "The average volume of API transactions for a Tier 1 wireless carrier will significantly increment over the next four years [added: from 2012 – 2016] eventually reaching 94 Billion transactions a month on average." That would be a lot for a single carrier, given that Netflix had 10 billion API calls a month in Jan 2011. Maybe things come together in the end …

References

[1] GSMA OneAPI: http://oneapi.gsma.com/
[2] GSMA OneAPI in Canada: http://www.slideshare.net/oneapilive/gsma-oneapi-gateway
[3] GSMA: http://www.gsma.com/membership/
[4] OneAPI API list: http://oneapi.gsma.com/api-list/
[5] AT&T developer program: http://developer.att.com
[6] BlueVia site: https://bluevia.com/en/
[7] Vision Mobile: http://www.visionmobile.com/product/developer-economics-2012/
[8] OneAPI Data Connection Profile API: http://oneapi.gsma.com/data-connection-profile-restful-netapi/
[9] BlueVia tour: https://bluevia.com/en/page/tour
[10] AT&T network APIs: http://developer.att.com/developer/basicTemplate.jsp?passedItemId=12500043
[11] BlueVia APIs: https://bluevia.com/en/page/tech.overview
[12] BlueVia payment API: https://bluevia.com/en/page/tech.APIs.PaymentAPI
[13] AT&T Device Capabilities API: http://developer.att.com/developer/basicTemplate.jsp?passedItemId=13100102&api=Device%20Capabilities&version=2&method=&provider=
[14] Apigee White Paper: "Is your API nacked? Roadmap considerations for API product & engineering managers" http://apigee.com/about/content/your-api-naked
[15] AT&T SDKs and tools: http://developer.att.com/developer/forward.jsp?passedItemId=100026
[16] BlueVia libraries to wrap APIs: https://bluevia.com/en/page/tech.libraries
[17] AT&T API console: http://developer.att.com/developer/basicTemplate.jsp?passedItemId=12700057
[18] AT&T links to best practice for apps programming: http://developer.att.com/developer/forward.jsp?passedItemId=7200042
[19] AT&T Application Resource Optimizer ARO: http://developer.att.com/developer/legalAgreementPage.jsp?passedItemId=9700312
[20] AT&T Developer Summit 2012: https://developer.att.com/developer/forward.jsp?passedItemId=9700278
[21] BlueVia Inspiration section: https://bluevia.com/en/page/inspiration
[22] AT&T Foundry: https://www.foundry.att.com/
[23] Telefonica's Wayra: http://www.telefonica.com/en/digital/html/venture_capital/wayra.shtml
[24] http://adage.com/article/digital/mobile-app-economics-fruit-ninja-makes-400-000-a-month-ads/235965/
[25] David Streitfeld "As Boom Lures App Creators, Tough Part Is Making a Living", NYT, November 17, 2012 http://www.nytimes.com/2012/11/18/business/as-boom-lures-app-creators-tough-part-is-making-a-living.html?_r=2&pagewanted=all&
[26] Pascal Finette: http://blog.finette.com/2012/11/20/the-false-economy-of-apps/

[27] Telco 2.0 Research on network APIs: http://www.telco2research.com/articles/AN_apps-apis-facebook-skype-imessage_Summary
[28] BlueVia business models: https://bluevia.com/en/page/biz.businessmodels
[29] Mind Commerce: Telecom Network APIs 2012 – 2016, http://www.mindcommerce.com/Publications/TelecomAPI_2012-2016.php
[30] Telenor joins BlueVia: http://www.eurocomms.com/features/analysis/8602-telenor-joins-bluevia-and-urges-other-operators-to-follow