# Modern Interactive Internet Services

Valentin Onossovski, Andrey Terekhov

## Web 2.0

After the autumn 2001 (famous ”DotCom Bubble”) crisis perspectives of  Internet and web-services were considered to be exhausted. But as usual only some specific technologies were really exhausted.

 30 of September 2005 is a birthday of new Internet paradigm - Web 2.0. The main Web 2.0 features are interactivity, web-access to databases and software as a service. Tim O’Reilly (http://oreilly.com/web2/archive/what-is-web-20.html) became the godfather of Web 2.0 and formulated the main principle “the bigger the number of users the better is information quality”. One more important feature of new approach is that the dialog with user does not obligatory start by his initiative. To define differences between Web 2.0 Internet applications and traditional ones O’Reilly gave the following example: personal homepage which can be read only –> blog where everybody can add his comments. This example perfectly shows the main idea of Web 2.0: interactivity and accumulation of information in the database.

As an additional example let’s consider one of the today’s popular services – traffic jam indicator. There is a mobile phone in a driver’s pocket, every second it sends its location. The information goes to central server which defines places with traffic jams (by average speed on every street).

One more Web 2.0 idea can be illustrated by a lot of peer-to-peer services, where each user acts not only as a receiver of some information but as a transmitter as well. This idea also works properly only after the critical mass of users is taking part in it and does not work at all with few users. The similar situation is with traffic jam indicator – it should be global concept accepted by many drivers.

We see that blogs have not so many differences with traditional Internet-forums (where replies, comments, pictures are also used), but examples like traffic jam indicator and peer-to-peer distribution are on the new level of user involvement.

One of the best examples is Wikipedia which content could be added by one user and be corrected by many others. These global systems could not be effective if only constrained group of users is able to change information. That's why such systems are hardly to be allocated in private corporate networks.

## Long tail concept

As it is known 15 percent of customers generate 85 percent of income. Another 85% of customers which generate 15% of income (usually called "long tail") was hard to be covered and many sellers ignored them – as they had to spend much more efforts to gain it. But nowadays when the Internet provides really global infrastructure and Web 2.0 ideas are widespread this long tail is more interesting for a seller. For example musical albums of not very popular singers often were not available because of seldom customers.  But nevertheless there is sufficient amount of people who could pay money for such music but could not find it. The first company that understood this was Google.

Considering existing popular Web 2.0 services, we can see that all of them are leveraging energy of people's social activities (communication, content exchange or even just driving in the city with a mobile device in the pocket) for improving their quality, replenishment of stored data and, finally, their expansion and growth as "ubiquitous" services covering more and more categories of users. So, high interactivity, mass character and the possibility of "almost infinite" growth are important features of modern Internet applications.

The new paradigm of large-scale, self-growing Internet services makes a fresh look at the traditional aspects of software engineering and software development process. For instance, what do notions like architecture, efficiency, reliability, security, scalability etc. mean for such class of applications? By figurative expression of Rick Kazman, one of the most renowned experts in the field of architectures of software applications, self-growing Web 2.0 services relate to the traditional software applications in much the same as the city relates to individual buildings. In our opinion, now it's a vast, unexplored and very interesting area that is still awaiting its researchers.

## PCs and mobile devices convergence

The trend towards globalization of modern Internet applications is in line with the global process of convergence between Internet, PC and mobile industries. There are a lot of PCs in the world but the number of mobile devices (MD) is at least 100 times greater. MDs (phones, smartphones, communicators, etc) have principle difference in usage against of PCs' usage (always in the pocket, always switched on, could indicate about outer events, could send its coordinates) and simultaneously have some important limitations (data input methods, screen size, radio channel capacity, battery supply, etc) therefore migration of software from PC to MDs is not so simple. Currently the largest part of information in Internet is formatted in accordance with PC requests (screen size and resolution, expected Internet traffic, etc.) while MDs impose very different requirements to the information. Today the most successful mobile online services are those that were initially created especially for mobile platforms (e.g. Google Maps, Gmail, Yandex Maps, Opera Mini, etc). Anyway we are sure that current global trend is convergence of PCs and MDs.

The goal is to expand PC technologies for wide population layers, providing online services that are currently associated with PCs for "non-computerized" users ("advanced housewives"). Simultaneously there is a reverse trend: migration of new ideas produced by modern MD services to the PC world.

Unification of service development for MDs includes 2 approaches:

- "Platform-centric" – development of services as native applications for each specific platform;
- "Service-centric" - creation of services on the base of unified middleware operating equally on different platforms.

Now platform-centric approach seems to be the main trend for modern expensive smartphones. Such native platforms as iPhone OS X, Google Android, Nokia Symbian and MAEMO, Microsoft WinMobile, Samsung BADA etc,. are fiercely competing on the market, providing mobile developers with development tools and IDEs as well as business infrastructure for distribution of the applications. The main advantages of this approach are rich functionality and advanced UI providing by native platforms. Two main restrictions are high complexity of native applications development (in spite of all efforts that platform vendors spend to make it more comfortable for "regular" developers) and isolation of native applications within their platform ecosystems.

The service-centric approach is more suitable for development of mass services that should be able to work on different types of mobile devices. A striking example of this model is Mobile Ajax technology. The main advantage of such approach is its universality; two main drawbacks – functional restrictions (especially in access to MD's native features) and possible high resource consumption as the price of unification.

## Ubiq Mobile platform

We in St.Petersburg State University are working on development of the universal platform for creation of mobile online services. The platform (called Ubiq Mobile) provides rich functionality, comparable with the level of functionality of Mobile Ajax framework. At the same time, its requirements to resources (mobile traffic, computational power) are quite low, so Ubiq Mobile-based services can work on wide range of MDs and in different network conditions including slow GPRS and EDGE connections.

The key idea of the platform is the use of terminal architecture, where all applications are running on the server and MDs are considered as remote graphical terminals. Data transfer between server applications and mobile clients is performing in graphical mode via proprietary binary protocol built over TCP/IP. There is no need for special preparation of images being transferred; only those portions of the image that should be visible on MD's screen are really sending to the client. Such approach makes the platform "lightweight" (because only simple terminal clients are running on MDs) and suitable for wide range of mobile phones, including cheap ones. Two main restrictions of the platform stem from its terminal nature: relatively slow static user interface (comparing with animated UI of native applications) and inability to work offline. But for certain classes of online services the platform is targeted to – interactive information services, mashups etc., – these limitations are not significant.

We expect that the new platform – lightweight, resource-saving, easy-to-program, easy-to use – will expand the range of both developers and users of modern mobile online services and make a new self-growing Internet applications available for new categories of users.