Stay Open.

A Technical and Policy Analysis of Interoperable Internet Messaging

Open-Xchange
Vittorio Bertola
vittorio.bertola@open-xchange.com

Version 1
September 2020

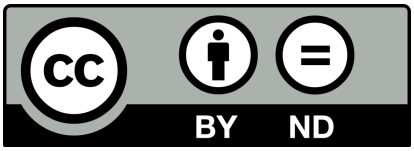

Table of Contents

1. Document scope 3
2. Executive summary 3
3. The concept of interoperable messaging 5
4. Technical architectures for interoperable messaging 7
4.1. Basic concepts 7
4.2. Software and protocols 8
4.3. Definition of interoperable messaging 8
4.4. Models for interoperation 10
4.4.1. Multi-protocol clients 10
4.4.2. Standardized server interfaces 11
4.4.3. Third-party bridging 12
5. Regulatory requirements for interoperable messaging 13
5.1. Scope of the obligations 14
5.2. Optional obligations for dominant clients 14
6. Public policy effects of interoperable messaging 15
6.1. User choice and competition 15
6.2. Accessibility and inclusiveness 16
6.3. Privacy 16
6.4. Encryption 18
6.5. Contacts' privacy and spam 19
6.6. Environmental effects 19
6.7. Innovation 20
7. A note on interoperable social media 20
8. Conclusions 21

1. Document scope

This document is meant as a contribution to the European regulatory debate on the introduction of required interoperability in dominant Internet messaging services that do not already offer it, and specifically instant messaging and social media. It will present some of the technical models that could be adopted, explaining them to a non-technical audience, and then discuss the public policy benefits and implications of interoperable messaging in general and of each model.

The following executive summary summarizes the analysis of the problem, of possible technical solutions and of the resulting policy effects, while the rest of the document explains and discusses each topic in detail. The last section provides final recommendations for regulatory action.

2. Executive summary

Two categories of Internet messaging services are broadly in use today. The first, email, is an example of an interoperable messaging system; users can pick one of many service providers, and the users of one provider can transparently exchange messages with the users of any other provider.

The second, instant messaging, does not however work in the same way, as most messaging services are closed and only available through a single service provider that controls all aspects of the service. Users do not have a choice and are forced to maintain multiple accounts on multiple systems, and to concentrate in those services where most users already are. As a result, the market is oligopolistic and hard to contest.

Social media is also, technically speaking, an extended form of messaging complemented by content hosting; and it also presents the same concerning situation of oligopoly as instant messaging.

Before examining the effects of interoperability requirements, it is necessary to define what interoperability means. In section 4, after explaining the basics of how messaging systems work, we introduce two different degrees of interoperability:

- **Full interoperability** allows the user of a messaging service to exchange messages with users of other messaging services as if they were users of their own;
- **Client interoperability** only allows the development of multi-service client applications that would allow the user to continue communicating separately with contacts on each messaging service, but within a single application.

Full interoperability also requires the definition of a set of interoperable features, which are to be supported uniformly by all messaging services (while still allowing each service to develop its own additional features), and of globally unique user identifiers that can pinpoint users globally across all services (possibly, as with e-mail, based on the Domain Name System).

We present three alternative technical models for interoperable messaging. The first, multi-protocol clients, enables client interoperability; the second, standardized server interfaces, enables full interoperability; the third, third-party bridging, would also enable full

interoperability, but with additional complexity and the need for a new type of players in the market.

In section 5 we will present two different interoperability obligations that could be inserted in upcoming regulation. Option A, a requirement for dominant services to publish their own interoperation interface and API, would establish client interoperability; option B, a requirement for dominant services to publish a standard interoperation interface and API and to interact with other services, supporting a standard set of interoperable features and standard globally unique user identifiers, would establish full interoperability. Option B would also require a technical policy process based on *"guided consensus"* to define and maintain the common technical standards. We also note how these obligations should only apply to dominant services that overcome certain thresholds of usership and relevance.

While our options for regulation only focus on the server side of messaging services, we will add that obligations on dominant clients could also be considered to enhance the pro-competitive effects.

In section 6 we will then analyse the policy effects of these options and the issues that they could raise.

In terms of competition, option A would create a market for alternative client applications but only offer more limited opportunities for competition among services, while option B would create a more competitive market for messaging services as well; thus we argue that full interoperability is necessary if user choice and competition for service is the objective.

Both options, by allowing the development of specialized messaging clients that better suit users with special needs, would be beneficial to accessibility and inclusiveness.

In terms of privacy, we note that the user's privacy is protected if the message is safely delivered to the intended recipients, and this is independent from whether the recipients use the same or another service provider, as long as providers are properly regulated and trusted by the communicating parties. In this respect, option B, by giving users choice on service providers, makes it easier for them to pick a party they trust; and by allowing the establishment of competing service providers, including providers located in jurisdictions with strong privacy protection (like Europe) and with business models other than user data monetization, incentivizes providers to compete on better privacy protection.

Encryption, as a technical device necessary to proper data protection, will not be negatively affected by interoperability requirements, as standard, public encryption algorithms are recognized as the most secure ones and can be supported by any interoperation mechanism.

Privacy also extends to contacts lists, and to the need to prevent spamming and undesirable messages; but these concerns can also be addressed in the design of the interoperation mechanism.

Both options would benefit the environment in some way, by allowing users to install and run a single client application rather than multiple ones, reducing the consumption of energy and technical resources especially on mobile devices.

Finally, we reject the concerns that strong interoperability requirements could stifle innovation; our full interoperability requirement would still leave dominant services free to

introduce non-standard, non-interoperable features, leaving it to policy discussions whether they should later be included in the required set of interoperable ones.

In section 7 we note that our analysis generally also applies to social media, encouraging regulators to extend interoperability provisions to that field as well.

In the end, we conclude that full interoperability is entirely possible and that it would be beneficial to European citizens and to the European economy, and we wholeheartedly support the introduction of technical interoperability requirements as described in our option B.

3. The concept of interoperable messaging

An **Internet messaging service** is a service that allows a user to send content – generally text, optionally with files, media and other content attached – to one or more other users. Internet messaging services are traditionally divided in two service categories:

- **Electronic mail**, where the emphasis is on the reliability of the delivery and on the ability to send big quantities of complex content, even if not in real time;
- **Chat**, later also called **instant messaging**, where the emphasis is on the immediateness of the delivery and on the ability to interact in real time with one or more correspondents, even if with shorter and less complex messages[1].

Social media, while mostly being a content hosting, aggregation and curation service, also generally include a messaging component, as a way to notify in real time the appearance of new content and as an additional communications service among users. This messaging component would be key to making social media interoperable as well – hence the relevance of this document to social media as well.

Each messaging service adopts a **messaging system**, a set of technical and functional specifications that define how the service works. Some messaging systems are proprietary and just used by a single messaging service, while other systems are used by many different services. In the latter case, the services sharing the same system could also be **interoperable** and allow communications across multiple service providers; or they could just use the same system independently, without interoperating[2]. Thus, **interoperable messaging** requires both a messaging system that supports interoperability and messaging services and service providers that choose to use that system to interoperate.

---

[1] The terms “chat” and “instant messaging” are now generally used interchangeably, but originally “instant messaging” was introduced for SMS-like, smartphone-based messaging services aimed at one-on-one conversations with personal contacts, while “chat” is an older term often encompassing one-to-many, room-based virtual discussions, including with strangers. In the end, both types of systems converged towards the same functionalities, though often with very different user interfaces.

[2] For example, Skype is a messaging service and also a proprietary messaging system, with its own unique technical and functional specifications; Skype (the company) is the only service provider for the Skype messaging system. On the other hand, Jabber/XMPP is a standard protocol and messaging system which is used by many different services and service providers, generally in an interoperable way but sometimes also to establish private messaging services that do not interoperate with the other services that use the same system and specifications.

The two service categories above are useful to illustrate in practice what *"interoperable messaging"* means.

Email is an example of a generally interoperable messaging service[3], as it has the following characteristics:

- There is just one public email system globally.
- Email users can acquire their service from any of a very high number of email providers, often with very different business and service models.
- Email users can move to a different email provider and still keep their contacts (though, if they do not own their domain name, this will entail a change of email address, due to the lack of email address portability; and email data portability is mostly not implemented yet).
- Any email user can generally communicate with any other email user, even if their email providers are different.
- Email users can generally choose which email application or interface to use for accessing their email service, and access it from any device.

Conversely, instant messaging (IM) was originally also born through interoperable services, but with the advent of smartphones it became dominated by non-interoperable, competing services. As a result:

- There are several (not one, not hundreds) instant messaging services and systems globally.
- There is generally only one service provider for each IM system.
- IM users cannot move to a different IM provider for the same system – they can only move to a different system, losing all their contacts and conversations in the process[4].
- An IM user can only communicate with other IM users of the same system and service provider; to communicate with users of other service providers, the user needs to create additional accounts on those other systems.
- IM users can only use the application and interface provided by the service provider and can only access the service from the devices that the service provider supports.

As an exception, there is an up-to-date interoperable instant messaging system today, Matrix, and there are still users on older interoperable systems like XMPP and IRC. These systems already work "like email", with multiple service providers and client applications.

There is no technical reason for this difference between email and chat; it is just the result of market dynamics and of business strategies aimed at market capture. The rest of this document will provide examples of how instant messaging services could be made interoperable and discuss the public policy issues and effects of such a change.

---

[3] It is however possible to deploy private email services that do not interoperate, as they do not allow exchanging email with "outside" users, while still using email as the messaging system.

[4] If the user's contacts are also subscribed to the new IM system, and if the two systems use the same identifier (for example, a telephone number), it might be that the moving user will find the old contacts on the new system automatically, and will be able to start new conversations with them. However, past conversations will still be lost, and anyway this only works under the two restrictive conditions above.

4. Technical architectures for interoperable messaging

4.1. Basic concepts

Each messaging system is technically different – but not too much. We will describe concepts that can apply to any messaging system, though different systems often use different names to refer to them. These concepts will allow us to discuss the different possible architectures for interoperability and their public policy effects.

Each user of a messaging system is associated to a unique **user identifier**, which allows the other users to identify them and send them messages. The user identifier can assume multiple forms; it can be directly related to the location of the user on the network, so that it can be immediately used to deliver the messages, or it can be an abstract identifier like a (phone) number or a nickname.

Messaging services also usually allow the definition of sets of users and user identifiers, called **groups** or **rooms**, so that a user can exchange messages at once with all the members of the group/room.

In many service architectures, the user is also associated to a **home server**, i.e. a server where their messages are delivered, and from where the user's client applications will retrieve the messages to show them. This allows messages to be delivered even when the user is offline or is using a device (e.g. a mobile phone) which has limited or private connectivity to the Internet and thus cannot be reached directly by the sender.

In closed systems, there is no choice of home server; the owner of the service will silently provide one. In open, federated systems, users pick a server of their choice; in this case, senders need to know which one it is, and this information is often stored in the user identifier[5].

In some "serverless" architectures, there is no home server, and messages can only be delivered directly to the user's client application, with increased privacy and simplicity, but also with the disadvantage that no messages can be sent while the user is offline. Even in this case, however, there often is a centralized **directory server** that allows senders to discover the user's current location. A directory server, or a distributed database like a blockchain or the DNS, can also be used in federated server-based architectures to discover the user's home server from an abstract identifier.

A few totally serverless architectures adopt **broadcast** models, i.e. send all messages to all users and each user picks their ones, but this is impractical on non-local scales.

The information on whether the user is online and their location on the network is generally called **presence information**, and it can also include the user's photo, profile etc. It is a key element of a messaging system, though its actual content varies a lot from system to system.

[5] This mirrors the email architecture, in which the "home server", i.e. the user's mail server where their mailbox is located, is specified in the email address by the email domain, the part after the '@' sign, which can then be converted into an actual server name and IP address via DNS queries.

4.2. Software and protocols

Messaging services are generally implemented through the combined use of two different types of software:

- **Messaging server** software, to be run on home and/or directory servers, manages the user's message box and presence information, and exchanges messages with client applications and, in multi-server systems, with other servers;
- **Messaging client** software, to be run on user-facing devices such as computers and smartphones, receives incoming messages from the home server, displays them to the user and delivers the replies back to the server - or, in architectures without home servers, directly exchanges messages with the client software of other users.

In closed systems, there is just one server software, and just one client application per device/operating system; they are all developed and controlled by the service provider and can speak any protocol, even unspecified ones (i.e. protocols whose specifications are not formalized and distributed).

In open, federated systems, there can be more than one server and more than one client; these systems usually adopt public, open protocols, so that each developer can write servers and clients that interoperate with those developed by other software developers and service providers.

As an intermediate approach, some closed systems may choose to publish a specification for the client-server protocol and allow the development of alternative clients, while still keeping control of the service and of the user base through a proprietary server.

Also, independent client applications may choose to implement multiple protocols and thus function as multi-service clients, as long as all these services allow independent clients.

4.3. Definition of interoperable messaging

What does it actually mean for two messaging services to be interoperable?

There can be multiple degrees of interoperability, but we will define the following two:

- **Full interoperability** requires that the user of one messaging service (*"home service"*) can communicate with one or more users (*"remote users"*) of the other messaging service (*"remote service"*) as if they were users of their own messaging service. Consequently, any messaging client is able to interact with any messaging server; users can switch from their current client to another compatible client, and service providers can pick and switch between any server software. Users can also move from a server to another, and thus from a service provider to another, without losing their message archive and contacts, as long as data portability is effectively implemented.
- **Client interoperability** does not allow actual communication between users of different services, but allows the development of alternative third-party clients for the same service, and thus of clients that can interact with multiple services at once. Users can use a single application to interact with multiple services, but within that application, conversations and contacts are still broken into a separate silo for each messaging service. Clients are interchangeable with other clients, but servers

cannot be interchanged with others. Also, users cannot move to a different service provider while keeping their contacts and messages, though they can create accounts on additional services *("multi-homing")* without having to install additional apps.

Interoperability across multiple services that use the same system is almost straightforward, but if the services use different systems, their technical and functional difference comes into play. Client interoperability is not too hampered by a diversity of features across multiple messaging systems, as the client can tailor the user experience for each system. However, full interoperability can be hampered by the fact that different messaging systems have different or incompatible features.

Thus, for full interoperability it is also necessary to define a **set of interoperable features**, i.e. features that both services are required to support in a shared way so that they substantially work in the same way (though, for example, the user interface through which they are presented to the user may differ substantially, and unique additional features can be added by each service on top of the standard ones).

Defining the interoperable features is a matter of technical policy, but we would argue that at the current status of the technology they should at least include the following:

- Accessing the remote user's presence information;
- Adding the remote user to the user's contacts list;
- Adding the remote user to a group/room on the home service;
- Being added to a group/room on the remote service;
- Sending to users and groups/rooms of the remote service messages made of any combination of multilingual text, audio, video and attached files;
- Receiving similar messages from the users and groups/rooms of the remote service;
- Getting confirmation of the delivery and display by remote users of a previously sent message.

Since full interoperability is meant to include client interoperability and allow the connection of any client to any server, the set of "horizontal" (service-to-service) interoperable features above needs to be complemented with a set of "vertical" (client-to-server) interoperable features, that should at least include the following[6]:

- Authenticating on the home service, including support for the most common two-factor authentication methods;
- Retrieving received messages and submitting outgoing messages for delivery;
- If the service archives the messages, allowing the user to manage such archive, make backups, delete old messages, export messages for data portability etc;
- Creating and managing groups/rooms on the home service;

[6] In technical terms, there are architectural options that could influence the definition in detail of the interoperable features. For example, the delivery of a message to a remote user could either happen directly (the user's client connects to the remote user's home server) or through the user's home server (the user's client connects to the user's home server, then the user's home server connects to the remote user's home server and relays the message). These details can however be agreed and formalized when defining the technical standard for full interoperability.

- Performing all the operations of the “horizontal” interoperable set with the other users of the home service.

Today’s messaging applications increasingly include not just the ability to attach audiovisual content (e.g. vocal messages or videos) to text messages, but the ability to perform real-time voice or video calls between contacts; some convergence between video-conferencing applications and instant messaging applications is already happening. From a technical standpoint, this capability requires additional, dedicated infrastructure and standards, and specific security and anti-spam considerations. Consequently, whether this capability should be included in the set of interoperable features – also considering the potential regulatory overlap with “plain old” telephony – is open for discussion.

Full interoperability also requires the introduction of **globally unique user identifiers** that can reference users with certainty across all the interoperating services. This is not particularly hard; in fact, this would be the messaging equivalent of an email address, and could even have the same form.

Full interoperability, in functional terms, does not require that any of the interoperating messaging systems are open or use open protocols; as long as they establish ways to exchange messages and information among themselves, each of them could then continue working as a closed service with a single provider and no alternative client applications. However, as described in section 5, we would recommend achieving full interoperability by requiring all participant messaging services to publish a standard interface and make it accessible to any third-party client, as this would enable much broader positive effects on competition, and ensure that new entrants can join the market.

As an example, we note that both email (through the SMTP/IMAP standards) and mobile telephony (through the GSM standards) are fully interoperable systems with multiple interoperating services. On the other hand, client interoperability was only widely seen in instant messaging in the 00’s and did not last long.

4.4. Models for interoperation

Interoperability among different messaging services can be obtained through several different architectures. We will now describe three alternative architectures and their features, using them as reference for the regulatory options in section 5.

4.4.1. Multi-protocol clients

In this model, interoperation happens in the clients.

All participating services are required to set up a public interface (either one for the whole service, or one on each server), and to disclose an API or protocol that clients can use to communicate with it. This can be the same API that the service already uses for communication with its own clients, thus not requiring additional development efforts by service providers; in any case, the API must support all the features of the service, allowing the development of independent clients that offer all the same functionalities of the “native” one. Services are then required to accept communications to this interface by any client, except for the possible refusal of connections from abusive clients (e.g. for spam).

Client developers then have to implement the API of each participating service; this implies additional development work for each new service that the client wants to support, especially if the service adopts a different system and API than the other services.

However, this also allows clients to tailor their implementation to the functionalities of each specific service; different services could have different, incompatible features, and clients could still support them all.

On the other hand, interoperation between services would in fact be limited; this approach would basically work like conflating all client applications side by side into one, rather than like having a single application that can effectively talk with all messaging services at once and all their users in the same way, independently from the service they use. A service could still behave like a client, connect to a competing service and perform some exchange of messages, but the lack of a uniform set of interoperable features and of a common technical standard would not allow this method to support all features flawlessly, or even to support them at all. In practice, this model can only achieve client interoperability, not full interoperability[7].

This approach moves all the complexity of the interoperation into clients. As such, this approach is less burdensome for service providers (except those that may still want to attempt service-to-service interoperation), but more burdensome for any independent developer willing to build an alternative client application to compete with the established, dominant ones.

In fact, it could easily happen that client developers would limit themselves to implementing the APIs for the few most used messaging services, avoiding the effort necessary to support additional APIs for less used services; this would hamper the chances of successful competition by new entrants and by services that do not already have a significant market share.

Also, this model does not free users from being locked into specific services if they do not want to lose their contacts and conversations. It just makes multi-homing easier by allowing users to manage all services from a single app, which is a practical advantage over having to install multiple apps, but not as big as being actually able to switch in full to a competing service provider.

As a consequence, if the objective is to create competition and opportunities for new services and applications, this approach seems to be less effective than others.

4.4.2. Standardized server interfaces

In this model, interoperation happens in the servers.

An open, standardized server-side API or client-server protocol is defined, and all participating services are required to set up a public interface (either one for the whole service, or one on each server) that can exchange communications using that standard. Services are then also required to accept communications to this interface by any client, except for the possible refusal of connections from abusive clients (e.g. for spam).

Service providers would then have to implement within their server software an additional software component that “translates” the messages, data formats and features of the standard protocol into corresponding features of their own service. This would only work

[7] One could indeed imagine technical expedients through which multi-service clients could act as “bridges”, relaying messages from one service to another, and emulating some sort of full interoperability. However, this would be an ineffective and incomplete way of implementing full interoperability.

for the standard set of interoperable features that is supported by the standard protocol; any additional features of the specific service would not be accessible through third-party clients. Service providers would also need to add within their server software support for the interoperation with users of other messaging services, such as (depending on the actual technical architecture of the common standard) connecting with other servers to deliver and receive messages.

Service providers would also need to publish a way to convert their own user identifiers into globally unique user identifiers in the standard protocol, so that their users can be identified uniquely throughout all interoperating services. However, service providers would not be required to disclose their own protocols or the internals of their system.

On the other hand, client developers would only need to implement the single standard protocol, which would enable them to interoperate with all the participating services.

As the client-server interface would then be standard and same for all, this model would naturally achieve full interoperability.

This approach is more burdensome for service providers, but less burdensome for client developers. It could increase the development costs for the creation of new messaging services; however, once the standard is created, new services would possibly just natively design their implementation according to the standard, without the need to develop additional “translation” elements. On the other hand, this approach would make it much easier to create new clients, fostering innovation, competition and diversity in terms of user-facing applications.

As a drawback, this model requires the agreement on a common messaging system and on a set of interoperable features; such agreement needs to be overseen so that it maximizes the public interest, as private-sector-led standardization processes could instead agree on a standard that favours certain dominant players. Processes to update the standard and the set of features would also need to be established.

4.4.3. Third-party bridging

In this model, interoperation happens in the network.

It requires the establishment of separate “translation” services (usually named **bridges**) that would be capable of “speaking” the different protocols and messaging systems for two or more interoperable services; the clients of one service, when needing to interoperate with the users of a different one, would connect to an appropriate bridge that would receive the requests according to the client’s own system, convert them into the system of the other service, and relay them to the other service’s servers – and do the opposite for the replies. Again, service providers would then be required to disclose their protocols/APIs and make an interface publicly available, like in the first model; however, only bridges, not clients, would connect to this interface.

In this model, the burden of interoperability is borne neither by the clients nor by the servers, but by a specialized party that develops and runs the bridging servers. This makes the model very flexible, as any interoperability issues can be solved by the bridging party, without the need to change clients or servers. This also makes all existing clients and servers almost immediately able to interoperate with very few modifications.

This model also achieves full interoperability, as long as all services support the same set of interoperable features and as they can be practically "translated" from one service's system into another's.

However, this model requires that a third party supports a significant cost for the development and operation of the bridges; it is unclear which business model such a party could actually have. One could imagine new startups offering bridging as an additional service; however, since messaging is generally a free service for the final users, it is hard to imagine that users would accept to pay a fee just for the interoperation mechanism. New entrants could rather be interested in supporting the costs for bridging with the dominant services[8], as this could substantially increase their adoption, but doing so through this model would impose the cost of interoperation onto new entrants.

In other words, if promoting competition is an objective, unless there is a public service entity willing to bear the cost of bridging on a mass scale, the bridging model seems limited to low scale interoperability experiments.

5. Regulatory requirements for interoperable messaging

As a consequence of the previous analysis, there are fundamentally two alternative legal obligations that can be imposed onto dominant messaging service providers to promote a certain degree of interoperability:

- **Option A (Client interoperability)**. Require dominant platforms to establish an open, stable interface to their service and to publish their own protocol/API specification through which third parties can interface with it (the *"multi-protocol clients"* model).
- **Option B (Full interoperability)**. Require dominant platforms to establish an open interface to their service according to a specific open standard and messaging system, and to add support for interoperation with other services throughout their server software (the *"standardized server interfaces"* model).

Option A should be enhanced by a completeness clause: the API would need to support all the features of the service, so that third-party clients can be an effective replacement for the native one.

Option B requires a process to pick or design the open standard that every service will have to support, the set of interoperable features and a design for globally unique user identifiers. To this purpose, we recommend a process of *"guided consensus"* in which the industry, with the participation of user and civil society organizations, is required to agree as far as possible on a proposal for a common standard – or more than one, if no consensus is possible – in a reasonable timeframe, while a regulatory agency with appropriate technical and policy capabilities then validates the proposal, ensuring that it meets the overall policy objectives and that the standard is effectively accessible to all

[8] This has indeed happened within past and current attempts to build open interoperable messaging systems, where members of the community have often volunteered to develop and run bridges towards other messaging systems, so to make the new system more useful. However, this has only happened on a small, experimental scale that could be managed only with volunteered resources.

implementers (including non-profit, community and open source ones). To this purpose, such standard should be available free of charge and unencumbered by any intellectual property and patent licensing costs and restrictions. Existing standards could already be considered, or existing Internet standard bodies could be used to develop these standards, though there needs to be care in ensuring that whatever process is chosen cannot be captured or disrupted by the dominant players.

Both options would also need the establishment of additional guarantees, such as the prohibition for service providers of refusing access to their open interfaces, except for a limited number of clearly defined cases (for example, abuse for security attacks, attempts at spamming or denial of service, etc.) with appropriate redress measures for contested cases[9].

5.1. Scope of the obligations

Both models require a definition of when a messaging service can be considered "dominant". Regulation could just refer to very high level principles, such as market share and number of users (across Europe or in individual countries), economic relevance (for example in terms of advertising revenues, for services based on that business model) and consolidation effects (messaging services native to dominant platforms, e.g. dominant browsers, dominant operating systems or dominant social media, should also be considered dominant). A regulatory agency could then be tasked with defining them further and solving any quarrel on whether a specific service is dominant.

At the same time, it is important to establish thresholds so that newly created, independent and niche messaging services can be developed without having to bear the interoperability obligations from the beginning.

5.2. Optional obligations for dominant clients

The two regulatory options above only act on the server side of messaging services; no obligation is foreseen for the current dominant clients, e.g. apps like WhatsApp or Telegram that most users already installed and learnt to use.

Some argue that even if you introduce full interoperability and allow people to communicate across messaging services, this will not be effective if the current dominant clients are not required to add support for such communication. The final effect could then be that a few users move to alternative services and a few other users move from dominant clients to new multi-service applications, but since the apps that people already use do not allow communication with users of other services and do not even show the fact that other services now exist, most users will just stay with the dominant service and client.

This could be countered with an obligation for any dominant client – defined through usage thresholds similar to the ones for dominant services – to implement support for communication with the users of any service that adopts the standard interface.

We think that this issue needs to be considered more in detail, and we leave it to further discussion.

---

[9] These guarantees could be part of this new platform regulation, or could be deferred to specific regulation dealing with spam or with privacy (e.g. the long discussed ePrivacy Regulation).

6. Public policy effects of interoperable messaging

6.1. User choice and competition

The lack of interoperability, together with the strong network effects inherent in messaging, is a root cause of reduced competition in the Internet messaging market. Users need to use the messaging system where their intended recipients are, and this reinforces and ossifies the position of the current dominant services, that can push users to adopt their own client applications at their own conditions. Any attempt to enter the market with a new messaging service is significantly hampered by the fact that its users will not be able to communicate with anyone, unless they can convince other users to join yet another messaging system. Also, any attempt to enter the market with alternative client applications, even if better and more useful, is made impossible by the fact that most messaging services only allow access to their servers from their own client application or website.

Moreover, most dominant messaging services today are provided by big Internet companies that also provide many other services; their dominance of the messaging market and the exploitation of the user base gained through such widely used services could reinforce their dominance in other markets and vice versa, creating anti-competitive network effects.

Drawing on past regulatory experiences (e.g. with mobile telephony), it is generally accepted that the introduction of opportunities for interoperability and portability across service providers reduces the advantage for dominant players and creates opportunities for more choice for the users and for more competition on the market; this is generally beneficial to society at large.

The two regulatory options detailed in section 5 differ on the degree of choice that they actually allow.

Option A (client interoperability) creates more choice in terms of client apps, and favours multi-homing, but does not address in full the problem of users being locked into specific services, nor that of promoting the creation of new competing messaging services. The lack of standardization of the server interface makes client development complex, and can push client developers to limit the number of services they support; the migration from one service to another is made harder by the lack of a standardized set of common features, and service providers can even try to exacerbate on purpose their differences to make moving even harder, or evolve their own server interface in ways that make third-party clients less effective. Since there is no direct communication between users of different services, the service provider that the user employs is still all-important in the ability to communicate, preserving vendor lock-in.

Option B (full interoperability) creates more choice not just in clients, but also in terms of servers and service providers. As there is a set of common features and a single common technical specification, moving from one service to another is much easier. The development of clients and of servers is easier, leading to more opportunities for new entrants in both niches. The evolution of the common technical standard happens jointly by all participants, avoiding unilateral changes to specifications that can create costs and break interoperation. Even with limitations and the need for constant regulatory oversight

and enforcement, full interoperability can be a stable market condition that maintains at least some degree of competition over time.

This is why we would suggest that regulation should focus on establishing full interoperability; at most, client interoperability could be a stop-gap measure that can be implemented immediately while the technical standards necessary to full interoperability are worked out[10].

As an alternative, full interoperability could also be obtained through the *"third-party bridging"* model, coupling option A with the definition of a common set of interoperable features and with the establishment of third-party bridging and interoperability services. As we noted, it is hard to imagine that users would pay an interoperability service fee directly when all the messaging services themselves are available for free; this would put the burden of establishing and funding bridging services on those parties who have the highest interest in that, i.e. the new entrants, rather than on the dominant messaging services, reducing the positive effect on competition.

Finally, we would also suggest that the data portability provisions of the GDPR are referred to and strengthened by the new regulations, as they are another necessary element to make service portability possible and allow users to move easily among interoperable service providers.

6.2. Accessibility and inclusiveness

An often forgotten effect of services that allow alternative clients to exist is that they enable much better access by those users that have special needs, for any of several reasons (disabilities, use of minority languages, etc.). While the specific efforts by some players are commendable, big service providers designing their client applications need to focus on streamlining the user experience for the majority of users, as this is what will determine the success of their service. On the other hand, many groups would benefit from the use of specialized, tailored client applications designed for their own specific needs; by enabling the creation of independent clients, these specialized applications are also made possible. Both options, A and B, would thus have beneficial effects for accessibility and inclusiveness.

6.3. Privacy

The privacy model for messaging is straightforward: each user of a messaging service expects that each message is only made available to the destination user(s) they selected. Any other party having access to the content of the communication, including the service provider(s), is unduly violating the user's right to privacy.

We start by noting that almost none of the messaging services in use today meets this model in full, as their provider, even when claiming to implement end-to-end encryption, usually has full access to each user's unencrypted communications (see the next section for a technical explanation).

[10] We however note that some fit-for-the-purpose open messaging systems already exist; it could be necessary to enhance them to accommodate the entire set of interoperable features, but the technical solution for full interoperability should not take years to develop.

There are thus three key requirements to preserve the user's privacy:

1. That the message is only delivered to the intended recipients;
2. That the intended recipients also respect the privacy of the message and do not spread it further unless authorized;
3. That the client applications and service provider(s) involved in the communication adopt reasonable security and data protection measures and do not watch or redistribute the message.

The second requirement is independent from the technology and its regulation, while the first and the third are directly connected to them; but all three points depend on the user's ability to trust both the recipients and the technology providers involved in the communication. So, how would this ability be affected by the introduction of interoperability?

While the recipients and their trustworthiness do not change, the introduction of interoperability gives the user a much better choice of technology providers. Indeed, if a user today wants to send a message to a contact residing on a specific platform, either the user trusts the platform, no matter what policies and privacy guarantees the platform offers, or the communication cannot happen.

With interoperability, the user will be able to choose a different technology provider (just for the client application, with client interoperability; also for the service, with full interoperability). This allows the user to gain back choice and control over which data processor will get their messaging data and to select providers that can be trusted, creating at the same time an incentive for providers to compete in privacy-oriented features that can enhance the user's trust in them.

It is true that, with interoperability, the sending users only control half of the communication, and, in comparison with today's closed platforms, an additional party comes into play; as the communication also relies on the service provider of the recipient, they have to trust that provider as well – sometimes without even knowing which provider it is. This is not different from what happens with email or telephony; when I call a friend, my communication also transits over the friend's mobile operator, which could attack my privacy or could use insecure implementations that allow data leaks and third-party attacks. This happens at any time without being considered a privacy issue in itself, due to the combination of regulatory oversight (imposing and enforcing privacy and data protection rules) and of trust in the recipient, which includes trusting the fact that he/she will have chosen a reliable provider.

Moreover, users could easily learn from their app which other provider the recipient is using, and if they dislike the idea of having their communications go through that provider, they could just stop the communication or invite their contact to move to a more trusted provider. Interoperability would give users the ability to communicate with users of other providers, but would never force them to do so if they do not want.

In the end, we argue that users receive better privacy from an ecosystem of providers and client applications that have to make compelling offers to users to gain their trust, rather than by a single all-in-one provider which they cannot choose and with which they have no negotiating power; so option B, despite a broader circulation of information and messages, would be the most beneficial to privacy.

The current situation is also made worse by the fact that many dominant messaging services are provided by companies that also provide many other online services, creating opportunities for detailed user profiling and tracking across multiple online activities; these companies usually have user data monetization as their core business model, creating an economic incentive for them to reduce the privacy offered by their messaging systems.

This is even more problematic for European users, if we consider that almost all current providers are based outside of the European Union and of its privacy-focused approaches to technology; if a user wanted to get their messaging services from a European company, they would have a hard time finding one their contacts also use.

In any case, additional guarantees could be introduced to improve privacy levels; for example, a common standard interface for instant messaging could include features that provide information on the provider's name and location and on the country it is located in, while users could instruct their client applications to avoid sending data to services located outside of the European Union, if they so wanted. This kind of meta-information could provide innovative opportunities for proper consent management and for full user control over the transmission of data. However, there are cases in which the service provider does not want to disclose additional information on who and where they are, for example for services aimed at protecting sensitive communications (journalists, whistleblowers, dissidents etc.) – so any metadata disclosure should be optional.

6.4. Encryption

Almost all messaging services today employ encryption to prevent access by third parties. Most also claim to provide *"end-to-end encryption"* – but in fact it is just managed app-to-app encryption, not actual end-to-end encryption.

In managed app-to-app encryption, the messaging application also takes care to create, store and manage the encryption keys used in communications; as a consequence, the encrypted channel only connects the sender's app with the recipient's app, and the application, and thus the service provider, has full access to the unencrypted content of any communication. In this case, the service provider (notwithstanding their claims) could in theory analyse the content of the message or even covertly relay it to its servers or to other parties. Also, app-to-app encryption mechanisms often leave the metadata of the conversation (sender, recipient, date etc.) out in clear.

In full end-to-end encryption, it is up to the users to encipher and decipher the messages using a separate cryptography application, or at least to create and manage the encryption keys, so that the messaging application only ever gets access to encrypted content. It is however very hard to design an end-to-end encrypted messaging application that is also easy to use for non-technical consumers; thus, app-to-app encryption is the model adopted by most service providers.

We do not want to enter here into the policy debate on encryption; the GDPR already mandates the use of state-of-the-art technologies to protect communications, and any further specification about encryption would be better dealt with in other regulations (e.g. in an ePrivacy Regulation). Still, we challenge any claim that the use of "end-to-end encryption" by a messaging service, when meaning app-to-app encryption, is an inherent guarantee of absolute privacy, especially when the service provider is also a company whose business model is based on advertising and user monetization.

However, for encryption to be possible and secure in an interoperable scenario, the encryption mechanisms used by the sender and by the recipient, and by their service providers, need to be interoperable as well. This is easier to accomplish in the full interoperability model, as the adoption of a single, common, open standard can include common ways to signal which encryption algorithm is being adopted and exchange the necessary cryptographic information[11]. Under these conditions, encryption will not be weakened or impeded by moving from a set of separate services to an ecosystem of interoperable ones.

6.5. Contacts' privacy and spam

Privacy of the communication itself is not the only form of privacy applying to messaging. Indeed, any communications system – even plain old telephony – is nowadays subject to spam, and to the harvesting of contacts to that purpose. Also, the contacts list of a user can reveal highly sensitive information; it deserves full privacy protection. Even the presence information needs some degree of protection.

When designing an architecture for interoperability, care should thus be taken to ensure databases of users and user identifiers are not made public and cannot be easily reconstructed from existing public information or by guessing, and that the presence information of a user is only available to their contacts. There are technical solutions to this problem, such as non-guessable identifiers, and cryptographic tokens that a user's client can distribute to contacts to allow their clients to access the user's presence information on their home server.

A basic measure to prevent spam is to only accept messages from senders that have been previously added as contacts. Email, however, does not have such a feature; this increased the problem of email spam, but also supported the use of email as a universal messaging system where anyone with your address can write to you.

In the end, we think any standard interoperable messaging system should support both modalities, and it should be a choice by each user whether they want to accept messages from unknown senders or not; also, a competitive service environment could push service providers to deploy reputation- and algorithm-based anti-spam features, like the ones that exist for email.

6.6. Environmental effects

While the environment may not be the primary concern in messaging systems, the need to install and run multiple messaging applications at once, instead of a single one, causes a waste of precious technical resources, especially on more constrained devices such as mobile phones. The increased use of energy, bandwidth, computational power and storage space contributes to the quicker obsolescence of the device and increases the environmental footprint of messaging activities, which becomes significant once multiplied

[11] In cryptographic systems, security is never obtained by hiding the encryption algorithm from public scrutiny or by adopting non-standard, proprietary algorithms. On the contrary, the adoption of well known, publicly tested, standardized algorithms strengthens security; the secrecy of the conversation is given by proper confidential treatment of the private encryption keys and secrets, not by the confidentiality of the algorithm itself. This is why the standardization of encryption mechanisms actually contributes to better security.

for billions of messaging users across the world. Both options would thus bring some benefit to the environment.

6.7. Innovation

Some parties claim that requiring that all service providers support a standard protocol and a predefined, common set of features impedes innovation and the diversification of each service that is necessary to healthy competition.

We reject this claim; even if the requirement for full interoperability (option B in section 5) were to be adopted, this would not prevent each service provider and each application developer from implementing additional features that are out of the interoperable set, and from using them to draw users to their service or app.

It would be easy (and we recommend so) to design any standard messaging protocol in a way that allows the unilateral deployment of extensions to support additional features, even before they are standardized or added to the regulatory set of interoperable features. In fact, this is how innovation has worked for email in the last decades.

Moreover, nothing would prevent the current dominant messaging services, which control both the client app and the server, from having their app connect to their server in non-standard ways through a separate interface, supporting even more non-standard features. It would be up to the process that defines the set of interoperable features to determine if and when these additional features need to be added to the set to grant all services access to them.

7. A note on interoperable social media

While this paper focuses on interoperable messaging, its principles and considerations would still generally be valid for social media as well. Social media services are generally closed and based on single service providers that control the entire service and do not interoperate with others; they present the same scenario, and the same drawbacks, as instant messaging.

As an additional concern, social media today is a vital instrument for socialization and information, and even for the forming of the public opinion which underpins every democracy. Preventing oligopolies and gatekeeping roles and ensuring as much choice as possible for the users is even more urgent.

Technical architectures for social media systems vary, but in general, the core of social media is about messaging; whenever a user creates a new post, the post is stored on their home server and either the entire post or a notification message is sent to a “mailbox” on the home server of their followers, where a web client or a client app can retrieve them for display. To avoid the unnecessary circulation of big media files, media attachments such as photos and videos are hosted in a single place, on the user’s home server or somewhere else on the service platform, and referenced in messages through links, so that clients can retrieve them only when necessary.

As such, interoperable social media would require a specifically tailored interoperable messaging system and protocol (such as W3C’s ActivityPub) plus some hosting components that each service provider would supply to their users.

While some of the details would depend on the specific standard that would be adopted for interoperation and would thus need to be assessed at the time of that choice, in general the same policy considerations made for messaging would apply to social media.

For example, in terms of privacy, each new social media posting either is public – and thus it is meant to be accessible to anyone, like a page on a website – or has a specific intended audience, becoming just like a message sent to a finite set of recipients; the audience can be restricted as defined by the poster independently from the service that each member of the audience is using. Furthermore, media content would generally not be hosted outside of the user's home service, and thus would not be subject to additional circulation or copying other than the intended recipients downloading it.

We thus encourage regulators to consider both messaging and social media when designing interoperability requirements, and to establish similar requirements for both types of service at the same time.

8. Conclusions

We have explained how Internet messaging systems work, introduced two different degrees of interoperability – *client interoperability* and *full interoperability* – and provided three different technical models for interoperation – *multi-protocol clients*, *standardized server interfaces* and *third-party bridging*. The first two models can be turned into two different interoperability obligations to be inserted in upcoming regulation: option A for client interoperability and option B for full interoperability.

By examining the policy consequences of the two options, we conclude that both options bring benefits, but option B brings more of them. In particular, option B (full interoperability) is necessary to introduce an opportunity for choice and competition among messaging services themselves, and not just among client applications. This would also benefit the privacy of users, by allowing the creation of privacy-focused, European operators adopting business models different than user data monetization. It would not stifle innovation or reduce data protection, but rather restore opportunities for new and better service to end-users.

We thus recommend that an interoperability obligation based on option B – a requirement for all dominant service providers to expose an interface based on an open, standard protocol, to accept connections to it by any well-behaving client, and to participate in a common scheme of interoperable features and user identifiers – should be included in upcoming regulation[12]. This obligation should also be extended to social media.

We suggest that obligations on dominant clients could also be considered, as an option to increase the positive effects on competition.

We also recommend that a technical policy process to define such a scheme is established, encouraging the participation by all existing and potential operators and by

[12] As a second-best alternative, regulators could consider establishing the obligations of option A together with appropriate measures to enact the "third-party bridging" model without requiring users or new entrants to fund its costs directly; this could be accomplished either through a neutral, publicly funded bridging service, or through incentives and funding for specialized startups.

other interested parties from the industry and civil society, under the leadership and the public policy oversight of an appropriate European regulatory agency.

We stand ready to participate in such processes and to continue contributing to the quest for a prosperous, privacy-focused and user-centric digital society in Europe.