Abstract: This article deals with the concept of ideas about communication between particular mobile devices. For this purpose it was necessary to design a new architecture called M client M client server. Basic architectonic aspects of this concept (related to not only mobile but to wireless networks in general) are described in the text. There are also described similarities with already existing principles, especially multi-agent systems which seem to be suitable supplementation while developing this concept. Supposed applications of this architecture are also mentioned.

Key-Words: hybrid client, client having variable thickness (richness), M client, peer-to-peer, mobile agent, Java, software architecture, J2ME, Android, Windows Mobile / Phone.

1 Introduction
While developing applications in mobile environment [1] we ask a question if a design based on already used concepts such as client-server architecture is as efficient in the mobile environment as elsewhere. One of the main problems of mobile (or wireless in general) networks is an instability of connection when mobile client can lose the connection with a server during performing an operation etc. This situation doesn’t have to fatal because the mobile client can use the possibility to communicate with another neighboring client, which could mediate the operation. This access is a partial shift from the client-server architecture to peer-to-peer architecture. We will describe this idea in more details in suggestion of M client M client server architecture.

At the beginning let us describe the terms with which we will work at the development of this architecture.

1.1 Thin client
Thin client is a computer system the functionality of which depends on the server with which it is connected via network. The principle of the thin client is an application layer of, which takes care of receiving inputs from a user and displaying results. The term „thin client“ can be understood in both hardware field where it represents e.g. terminal (screen and keyboard) and software field. Here it represents a presentation layer of the application that also communicates with the server and doesn’t perform any application logic. In its consequences such application is impossible to be used in case of server or communication line failure or drop out.

The advantage of the thin client is that it can be relatively modest in hardware because all work is done by the server. On the other hand in case of the server failure such client stops working.

1.2 Rich client
So-called thick or rich client is in contrast to the thin client an application which doesn’t only rely on the server when performing the application logic but in a certain degree it can work without the server because a part or a whole logic of the application is included in the client.

This is to some extent an advantageous and reliable solution because the server failure does not stop working of the system entirely. The disadvantage is bigger requirement for client hardware both CPU power and storage capacity demands.

1.3 Hybrid client concept
An architecture of the hybrid client in contrast to the thin or thick client does not directly depend on the application logic of the client. The client which plays two roles becomes the hybrid client. The first
role is the role of a common client. To this role we can engage both thin and thick client according to a demand of an application logic.

The second role is the role of the server. The hybrid client can also play the role of the server. In such a case it must also implement basic server mechanisms. In a classification of clients the hybrid client resembles more to the thick client because it contains more than only presentation layer. Such a client is of use especially in mediation of communication where it acts as a mediator between other two parts of the system.

In this way it is possible to effectively solve the problems of for example gaining data from the server without the necessity of being in permanent connection with the server (consider signal coverage, weak signal with failures etc. in wireless networks).

Server features of the hybrid client can help with these issues. The hybrid client can keep the forwarded data in its memory and in a moment of the stronger signal it can transport the data to the server. When the server is unreachable due to e.g. failure of a part of the network the client is able to set a connection with another client that has an access to the server using a different type of connection (e.g. WiFi instead of GSM). Via such a client it can enforce saving data on the server or retrieving data from the server without currently having an access to it directly through the network. Similarly the hybrid client can serve as a cache containing the most frequently used data from the server which it will be able to make available for neighboring clients in case of network failure.

1.4 Concept of client having variable richness
Because the mobile device has usually performance constraints, it is more advantageous to operate there such applications which we can mark as the thin client. If this client also plays a role of the hybrid client we get into quite a paradox situation. The thin client will hardly implement the server features in a way to meet the devices’ performance constrains.

One of the possibilities how to solve this situation is to set a certain distribution of performance between clients. This means some mobile devices would constitute only a thin client and the others of only a hybrid client. Logically there is a question how to suitably establish such distribution in advance, especially in networks where frequent changes in location of clients occur.

The second possibility is to establish the concept of so-called client having variable richness. This client will act as the thin client but of course, to be able to grow it will have to implement proper mechanisms for this purpose. The client having variable richness is the thin client in an initial position with a simple implementation of a so-called growth factor. In a certain moment the thin client can become the rich – hybrid – client. After this the thin client that has just transformed into the hybrid client can offer its services of the server.

We can imagine such situations in the moments when there is an instability in one part of the network and at the border of a stable and an instable area there will be only thin clients available. Then the demand to grow rich can be sent to them from another client in the instable area so that this instable area keeps in touch with the server via this transformed client which has an access to the server and can even itself happen the server, eventually WiFi access point. Later when the need of hybrid client is over (e.g. connectivity restoration in a certain network region), it can lose its parts implementing the features of the hybrid client and get into the previous form – the thin client.

1.5 M client
Based on the previously described things let us characterize an M client. The M client is a mobile client, an application which can be operated in the mobile device. From a design point of view it is the hybrid client having server features and with the ability to become an autonomous unit in a certain part of the network. Considering performance constraints of some mobile devices the M client can be used as the thin client with the implementation features of the client having variable richness in the way that the thin client can anytime become the hybrid client (and reversely, of course).

1.6 M client and mobile device
Before it is possible to suggest the M client we must consider the possibilities of current mobile devices. We can divide the mobile devices into two basic groups: With Java support and without Java support. Considering the planned technologies (mentioned later) it will be very problematic to develop a proof-of-concept version of the M client for devices without Java support. For this reason we will not mention such devices in this moment and we will talk about the devices with Java support. We can divide them in a following way:

- Devices with J2SE support
- Devices with Java ME support (formerly J2ME). These are devices able to run so-called midlets. The midlet is the mobile
application usually based on Mobile Information Device Profile, either version MIDP 2.0 or newer MIDP 2.1. Limitation is a problem of this profile which belongs to Connected Limited Device Configuration (CLDC). CLDC is determined especially for the cell phones and PDA which don’t have a big memory. Most producers of cell phones today implement this CLDC configuration which slightly interferes in J2SE and use only a part of this basic Java platform.

- Devices with own (bytecode incompatible) JVM – Android Platform

Mobile devices with proprietary system usually have at least an implementation of CLDC configuration, as mentioned above.

Android is another platform. The system is based on modified Linux kernel running its own JVM (Dalvik), not compatible with Java on bytecode level. But as the Java language is used to develop application, we consider Android “Java-based”.

In a conclusion it’s possible to run the M client there where we have at least Java ME. The question is if Java ME is technologically strong enough for an implementation of the hybrid client. Regarding the boom of smartphones with Android platform it seems to be more useful to implement the M client right for this platform without using Java ME.

2 M client M client server architecture (M client P2P server)

Regarding the fact that this architecture combines principles of peer-to-peer architectures [2] and client-server architectures it was temporarily called an M client P2P server architecture. Later the M client’s part was fortified as the most important component of the whole architecture project. That’s why the name M client M client server architecture is more suitable. In the text below there is a description of the architecture. We will also describe the similarity of this architecture to other principles which could be used in the development of this access in mobile networks.

2.1 Architecture description

The questions of two mobile devices’ connection can be solved in a following way. Server will exist in a system and the mobile device will log in this server joining the network. After it will download the list of other connected mobile devices including their service description and information necessary for connection with such a device.

Newly joined device provides such information to the server. Every device will be responsible for a regular updating of its list from the server. This way can also be combined so that in case of a big server load these devices send the list of available mobile devices among each other.

Such architecture as mentioned above combines client-server architecture and peer-to-peer architecture. Simultaneously the mobile client can in some extent behave as an autonomous agent. The mobile client is called the M client. We could call the above mentioned architecture „M client P2P server“ or better „M client M client server“.

![Fig. 1. Simplified suggestion of M client M client server architecture (M client P2P server)](image)
The aims of the agent can be proper or can arise from the other agent’s requirement – the agent with its own aims is analogous with the mere client, the agent accepting requirements of other agent is analogous with the hybrid client. Despite the suggested architecture contains the server it is in a boarder context possible to mark it as decentralized (like the multiagent system). Decentralization means the hybrid client can behave as the server and in some cases it can fully substitute this server. We can further generalize the architecture (and approach to the multi-agent paradigm) to peer-to-peer network of the hybrid clients – agents in an analogous way – that use the services of other clients and themselves offer some services to the others. Mobile agents are an extension of the multi-agent paradigm, i.e. software agents the code and data of which it is possible to migrate among nodes or locations. Mobility of the code can be used for an implementation of the client having variable richness.

There is a possibility to implement this architecture above an available framework for multi-agent systems. The requirement for a multiplatform application and support of the mobile devices are fulfilled among others by JADE framework for Java platform [3]. This can also be recommended because it is in the long term maintained, well documented and distributed as an open source [4]. A disadvantage of JADE is the fact that its architecture is not fully decentralized from the network communication point of view, but on the other hand it is possible to avoid the single point of failure using replication of the main container’s functionality into more containers. As a source of information about other nodes we can use an build-in DF agent of the JADE platform which already acts this role for the needs of the platform itself.

2.3 Communication barriers in mobile peer-to-peer environment

Mobile environment or in more general wireless network environment brings some specific problems related to the lower reliability of transfer medium, restricted bandwidth and restricted resources of an end-point devices. These complications and their solutions can influence the architecture of the final communication infrastructure.

The first barrier in direct communication is represented especially by firewalls and network address translations (NAT). Communication between shaded nodes must then be mediated via another node – called an active node, a super node or more generally the server. For increasing the effectivity we can use some sophisticated techniques for passing through the firewall, such as UDP Hole Punching [5]. But the use of the super node decreases the decentralization of the whole system and for increasing the reliability and the scalability it is necessary to have a choice of more mediators.

In case of mobile devices the possible change of an address of the mobile node during its movement is another problem. Again the solution is to use the mediator which is represented by a „home node“ of the mobile node and which mediates the communication with it.

JADE solves these problems with the aid of LEAP add-on[3], which enables to divide the container of the mobile device into two parts – a FrontEnd and a BackEnd. The FrontEnd is run in the mobile device and the BackEnd represents the „server part“ of the container, which is run in the stable device. The connection between the FrontEnd and the BackEnd is always made from the FrontEnd towards the BackEnd. This ensures a reliable transfer through the address translations and firewalls. This implementation also solves the question of the re-connection in case the previous connection was accidently terminated and of the cache for a temporary placement of undelivered messages. The disadvantage of the whole solution is the direct dependence of the mobile device on its BackEnd (this cannot be simply replicated for increasing its reliability).

As an alternative for transparent communication a solution based on VPN can be used. An open-source implementation N2N [6] seem to be very promising. It is L2 VPN with the support of peer-to-peer communication. The VPN server (the super node) serves mainly for the initial gaining information about other nodes. The communication between nodes passes over the server only in case when via no available technique it is possible to create direct connection between nodes. N2N supports the increase of reliability with the aid of replication of the super node’s functionality into two nodes. Unfortunately N2N is not fully scalable so that any node can represent the mediator of communication – the super node. We must mention that the super node does not have to directly correspond with the hybrid node from the suggested architecture’s point of view.

In case of use of the architecture in the way the agent-based approach is utilized it is possible to combine e.g. JADE framework with N2N VPN. VPN would ensure the transparent communication for JADE agents.

JADE fully supports so-called mobile agents[3], that is the possibility of code and data (agent)
transfer from one node to another. It is possible to use this for the concept of variable richness.

Another possible communication platform can be represented by one of the transparent peer-to-peer networks’ implementation, such as JXTA for Java. This one compared to N2N has more complex architecture (among others more types of nodes) which enables a bigger degree of decentralization. JXTA also has some features of an agent-based framework.

2.4 Character of mobile and other wireless networks

Although the M client M client server architecture was originally designed for mobile networks it is probable that it’s applications could be also used elsewhere. Community wireless networks are examples. Although these networks are mainly fixed they have many common characteristics with mobile networks at a closer examination. Most nodes are represented by fixed end-point nodes wirelessly connected to a bone network which is from a big part composed of wireless point-to-point connections. Some networks support the mobility (or if you like roaming) among particular access points – in such a case the character of the network is even closer to the mobile networks.

There are some common features of the community wireless networks and the mobile networks – such as possible failures due to disturbance, worse signal availability etc. Then it commonly happens that some nodes are unavailable due to leaving the covered area of the certain network or switching-off the node (a cell phone, PDA, personal computer). Typical barriers in the transparent communication such as those mentioned above (e.g. NAT, firewalls) can also be common. The environment of community wireless networks as well as the purely mobile network environment are obvious candidates for an application of the suggested architecture.

3 Supposed architecture applications

In the following text we will outline the supposed applications of the suggested architecture which could be a motivation for its introduction.

3.1 Distributed supportive information system of wireless network

There is a possibility of using the M client M client server architecture for the support of mobile network’s operation. Especially in case of ad-hoc networks or other spontaneously created networks the possible supportive communication and information system could be a benefit. Its aims would be the following – assistance to a user in solving communication problems, preparation records for its possible analysis by an expert, improvement of user’s knowing about the state of the network (planned drop-outs). There are many instruments for achieving these aims, that are out of scope of this paper. Let’s describe one scenario.

There is a concrete use case, which is a planned network drop-out. A node couldn’t have been informed about this because it had been out of reach of network or switched off before the drop-out was planned. After its joining to the network it is not able to communicate with the central node which had originally propagated the information. Here it is possible to use the M client M client server architecture for the node to find active and available neighboring nodes and to ask them for an information about the failure. The condition is these nodes remember (cache) this information. The result is the user is correctly informed about the planned drop-out, including its duration.

Final information system could be described as a distributed database system or eventually a peer-to-peer database[2].

3.2 Mobile community network

The internet access while travelling i.e. the mobile access still stays a very costly service. As an example of world-wide community based network we can name FON based on WiFi hotspots. But in an idea of mobile community networks we can go even farther. Local mobile providers (e.g. GSM) commonly offer an affordably priced service of mobile internet for a fixed price (flat-rate) to their clients.

In contrast to this the mobile internet access in roaming is far too expensive or the maximal amount of transferred data for a fixed price is very limited. If a traveller liked for example to send an email from a smartphone abroad without having extreme financial expenses he would be forced to find an available hotspot. But if we realize that around him there probably are customers of local mobile providers who already have a mobile internet access in their smartphones but don’t use it right now, an idea about an advantageous cooperation occurs.

With the aid of the above suggested architecture we can to develop a mobile application for smartphones that would technically provide the traveller’s phone’s connection with the phone of a local user (on a voluntary base of both). This could
be reciprocally ensured by participation of both in a „mobile community“.

3.3 Cooperative navigation in buildings
Location based services are more and more popular. But GPS has got its limits, especially inside buildings. Vendors react by employing other ways of localization such as those using GSM cells or WiFi access points. Nevertheless only WiFi-based localization is applicable inside buildings, but still it needs a database of particular access-points’ locations. Their density and suitable positions (e.g. in department stores) will not probably be sufficient for a very precise orientation.

There is an idea of the use of this architecture for cooperative navigation of mobile devices. It is obvious that the mobile device which has been moving in e.g. department store for some time already knows the strength of signals of each WiFi APs in different places of a building and thanks to this it can help to a newly coming device in better orientation. This concept can be further extended in function of another „light-house“ (beacon), a mobile device that already knows its exact position and can play the same role as a fixed AP for other mobile devices – this means to be a source of signal for triangulation.

3.4 Communication support in disaster areas
Ensuring communication support in disaster areas can be another possible application of M client M client server architecture. In these areas the mobile signal failures can occur or somewhere it can even be entirely unavailable. Thanks to this architecture it would be possible to ad-hoc mediate communication among the operation center, rescue squads and inhabitants immediately after the disaster where there is not the full function of mobile networks restored yet.

4 Conclusion
M client M client server architecture is now rather an idea which seems to be very promising for future utilization in communication among mobile clients. It combines advantages of two architectures i.e. classic client – server and peer-to-peer architecture. The client – server mode with central (eventually replicated) server will be used in an ordinary conditions when the server is available for the M client. In case of the server unavailability the M client will be able to switch to the peer-to-peer mode and try to finish the desired operation in cooperation with other available nodes.

It seems it will be useful to explore components of multi-agent paradigm which offers already existing implementations of agent frameworks. These could be used as a base of implementation of the suggested architecture. The aim is to create a real implementation of such architecture in the future, test it and based on this keep improving the concept. The architectonic concept M client M client server could in the future become another pattern in the development of wireless networks, services and their utilization not only in company field. Supportive information system of the wireless network, mobile community network, cooperative navigation in buildings or communication support in disaster areas are examples of the use of this architecture.

References:

Acknowledgement:
This article was supported by the project No. CZ.1.07/2.3.00/20.0001 Information, cognitive, and interdisciplinary research support, financed from EU and Czech Republic funds.