Mesh network

I’ve finished research regarding of building mesh networks on mobile devices and came to the following conclusions:

Prototype development

1. The iOS operating system on Apple smartphones doesn’t allow to build a mesh network in the regular mode of operation. And Apple doesn’t allow to control devices at a low (“core”) level, so it is impossible to switch device’s wifi interface to mesh point mode to create or join mesh network.
   Also, it is impossible to build a mesh network between Apple devices and Android devices at the moment, due to Apple uses the closed protocol “MultipeerConnectivity” on their smartphones that supports in iOS operating system only.
2. The Android operating system doesn’t allow switching wifi mode to mesh point mode in regular mode of operation, but this restriction can be bypassed using root access. To switch between managed mode (default wifi mode) and mesh point mode I use the “nl80211”, “netlink”, “genetlink” libraries and system calls. We must have CAP_NET_ADMIN rights at the operating system level to make changes to the configuration of Wifi hardware. In Android operating system (and in all Linux operating systems) only the “root” user has such rights, and there are no possibilities to give CAP_NET_ADMIN rights to another user.
   In order a developed program has the CAP_NET_ADMIN privileges at the operating system level, we should develop a small kernel patch (just several lines of code), and this way will allow us to run programs with the CAP_NET_ADMIN privileges and root access will not be needed. But the use of such a patch on Android devices, in fact, is a “JailBreak”.
3. Wifi module in the mobile device must support the mesh point mode on the hardware level. In some type of devices wifi module supports this mode on the hardware level, but vendors do not provide appropriate driver which supports mesh point mode at the operating system level. In this case we should implement corresponding driver ourselves.
4. I see the next option how to solve this problem - sell patched Android devices (“Smartmesh”phones). And to make the source code of core and patches open for the community to be able to verify the safety of these devices.

Implementation plan:

1. Develop a patch for core of Android operating system (version of Android 5 and higher)
2. Develop a library for switching wifi modes and for mesh point configuration.
3. Develop a test (“one-button”) application for Android which can send and receive raw frames to and from the mesh network, for demonstration.

Estimation to develop a prototype: ~ 500-650 hours.

Mesh network protocol

It is necessary to develop a graph-based network protocol for the interaction between devices in the Mesh network.
What we need to develop a network protocol:

1. Implementation a Gossip protocol for the mesh node. Each node will have a unique ID formed by a special algorithm, which is used to form private keys in Ethereum.
2. Develop a model and workflow of messages reception and transmission in mesh networks: ping, pong, transit message, etc.
3. Develop a module for GET and SET operations to work with a key / value database to store the current state of the mesh network.
4. Develop an algorithm to find the shortest path between two nodes, based on local data in the database.

Estimation for protocol development is ~ 350-400 hours.

Atmosphere blockchain

I researched the code base of several blockchain projects, which can be used for Atmosphere development and I settled on the Hyperledger Sawtooth framework.
Out of the box, we get:

- EVM (Ethereum Virtual Machine) to support Ethereum smart contracts.
- JSON-RPC Ethereum-like API through which the smartmesh application will interact with the Atmosphere.
- The ability to create customized transaction families relatively quickly.
- The ability to change the consensus in the already launched blockchain, if we want to add new opportunities after the launch of the blockchain.
- The ability to implement a customized p2p mechanism which will provide communication between nodes in the mesh network (relatively easily, in comparison with Ethereum and Bitcoin).

What we need for Atmosphere blockchain implementation:

1. Adapt the Atmosphere blockchain based on Hyperledger Sawtooth with the Android operating system (version of Android 5 and higher).
2. Add master nodes functionality to the consensus protocol to encourage honest master nodes or punish those who want to deceive the network.
3. Create an AAR-library for the Hyperledger Seth transaction family, using gomobile.
4. Develop a small JSON-RPC API for the Hyperledger Seth using Go and integrate it to the corresponding AAR-library.
5. Develop a new Hyperledger Seth transaction family for communication between Atmosphere nodes using the mesh protocol.
6. Develop a Plasma smart contracts for EVM.

Estimation to develop Atmosphere:
~ 650 - 800 hours
We can use several developers to distribute tasks: Plasma, Atmosphere, master nodes.

Cross Chain

I’ve researched various possibilities for the interaction between blockchains. At the moment most perspective is to implement the cross-chain functionality as a centralized service. Person who want to make an exchange between SmartMesh blockchain and some other blockchain will have to transfer funds to the one-time address belonging to the cross-chain service and the service will in turn transfer to this person the equivalent of the cryptocurrency in target blockchain. The reverse exchange will work in the same manner.
What we need for cross-chain implementation:

1. Develop the architecture of infrastructure for the cross-chain service and corresponding deployment scripts (for chosen cloud solution, e.g. AWS CloudFormation in case of AWS cloud provider).
2. Develop the cross-chain service (backend) as a set of distributed modules (micro-services architecture). The interaction between the core module of cross-chain service and a specific supported blockchain will be provided by a separate module which from one side integrates with specific blockchain, from another side implements a common interface for the core module of cross-chain service. Thus, it is easy to add the support for a new blockchains.
3. Develop HTLC smart contracts for SmartMesh blockchain. SmartMesh blockchain participants will be able to execute exchange of different values with each other within SmartMesh blockchain using such HTLC smart contracts.
4. Implement the corresponding API for third-party services.
5. Launch (with redundancy) a full nodes of supported blockchains.
6. Optionally (does not included into estimated time), develop an web-portal with user-friendly interface for deposit funds and withdraw funds to and from the SmartMesh blockchain using an one of the supported cryptocurrencies.

Time needed for development:
~ 350-400 hours (most of the time it will take to implement the backend).