Design and Implementation of Effective Monitoring Solutions for Heterogeneous Wireless Networks

multi-network overlay construction

Group 3

Ao Shen(93564753)

Bichen Wang(75524768)

Chen Chen(31803753)

Hao Lian(20362736)

INTRODUCTION

- Multi-network management system
  - Server-client system
  - organizes different kind of networks and collects global status of the system in a logically centralized entity intermittently.
  - Based on the collected information, system makes decision to redesign topology and layout of the system.

- Apply overlay manager on server and clients

INTRODUCTION

- Overlay construction (main job)
  - Make multi-network overlay construction algorithm more robust, dynamic and scalable
  - Apply it to multiple network environments
  - Tools: JAVA

RELATED WORK

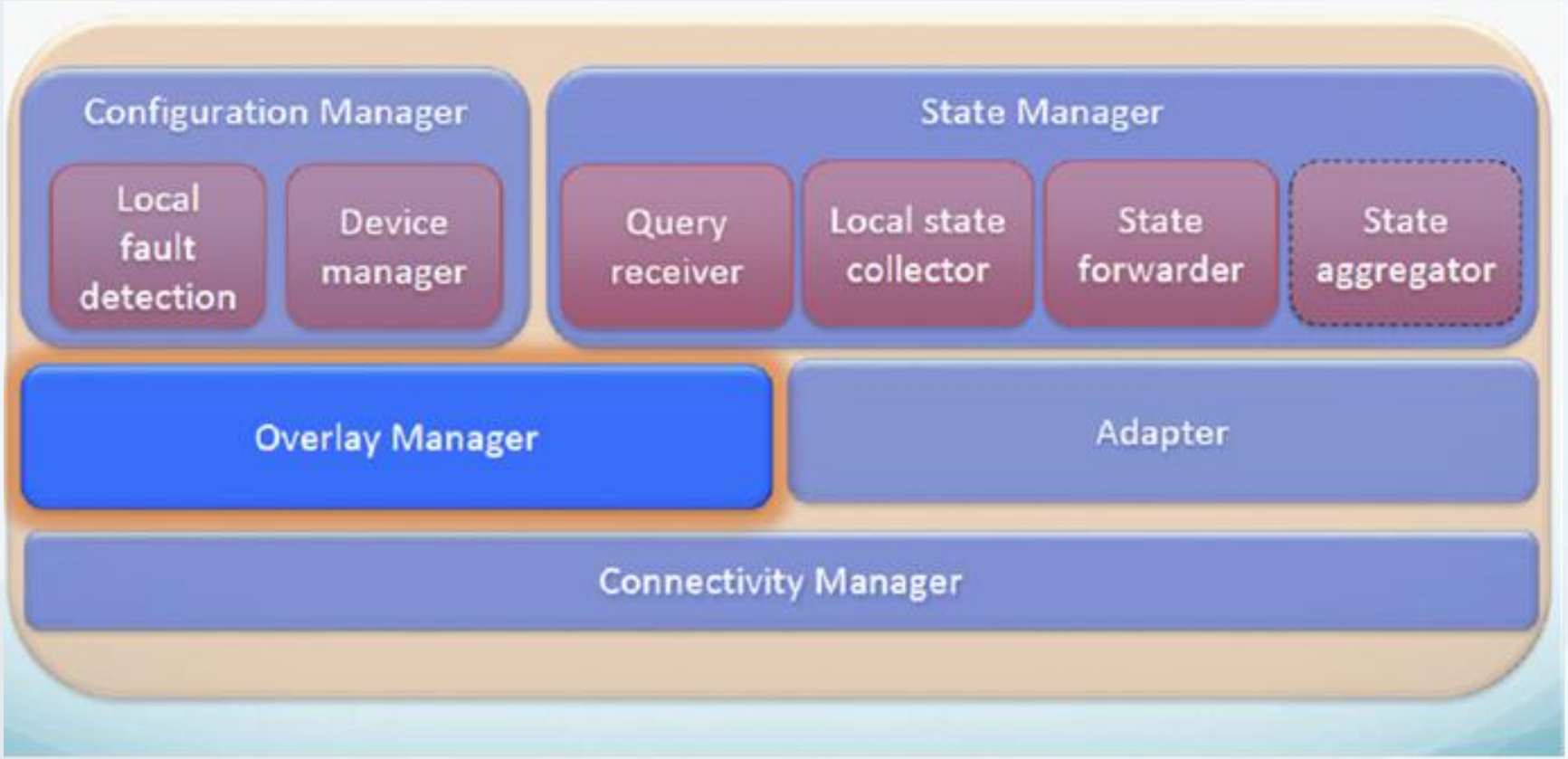
- Devices have different connectivity opportunities need a general mechanism to manage them
  - existing works manage specific networks independently
- Multinetwork as a composition of managed heterogeneous networks
  - instead of multiple independent managed networks
- Tighten integration to improve reliability and scalability
  - leveraging on network replication

CONTRIBUTION

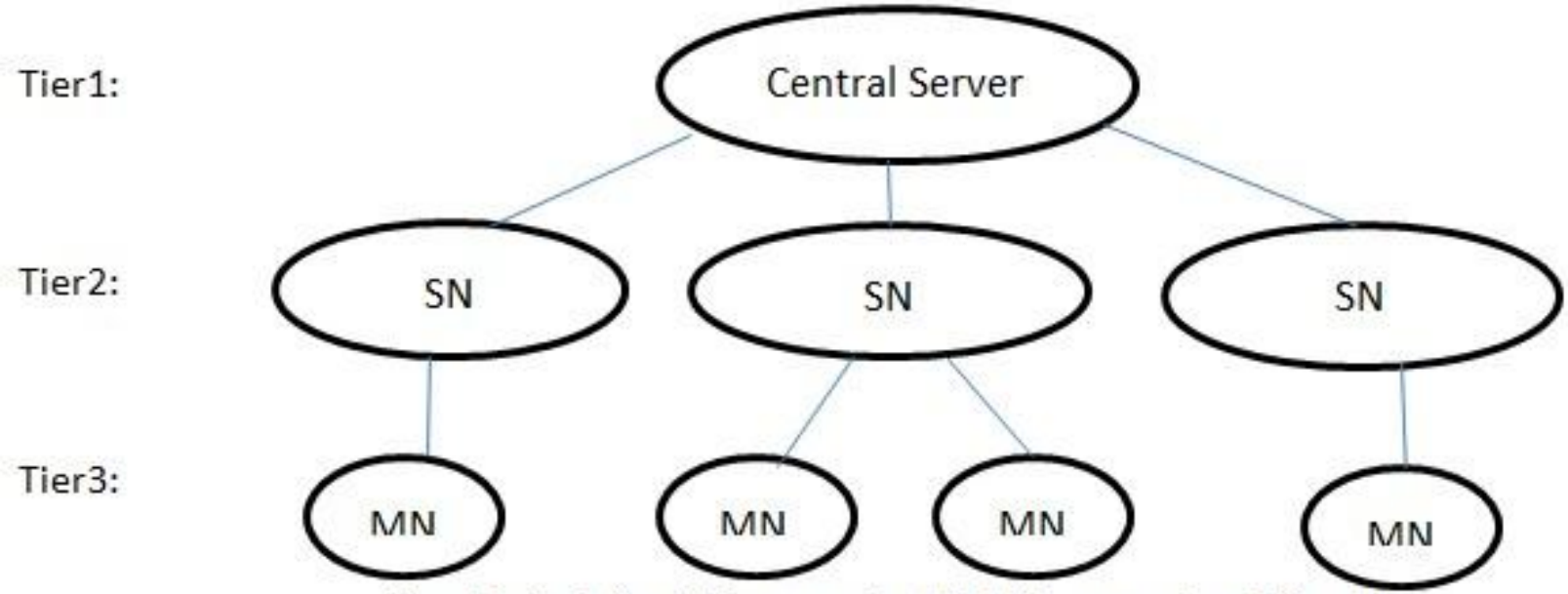
- Hao Lian is responsible for network configuration coding
- Ao Shen is responsible for module coding like the nodes, and timers.
- Bichen Wang is responsible for testing and debugging
- Chen Chen is responsible for testing and debugging
- We all participate in the discussion each week and contribute to the documents such as survey paper.

ARCHITECTURE -- Server

ARCHITECTURE -- Client



TREE-BASE DATA STRUCTURE



Here SN stands for stationary node while MN represents mobile node

DATA STRUCTURE

- Two types of nodes: server and client.
- Server
  - Only one server in this system
  - Server has children, but no parent.
- Client
  - client can be varies, from tier 1 to tier N.
  - Client have several children nodes or not, but only one parents, if any.

- Set of neighbor nodes can collect all the information of their neighbors, whether parent node or child node.

DATA STRUCTURE

- Two timers
  - One is for broadcast, responsible for broadcast every certain time.
  - The other one is for checking, responsible for checking if the parent nodes are connecting to the children nodes.

DATA STRUCTURE

- Six messages
  - Broadcast messages
  - Parent_request messages
  - Parent_confirm messages
  - Descendent_report messages
  - Parent_genchange messages
  - Parent_failed messages

ALGORITHM

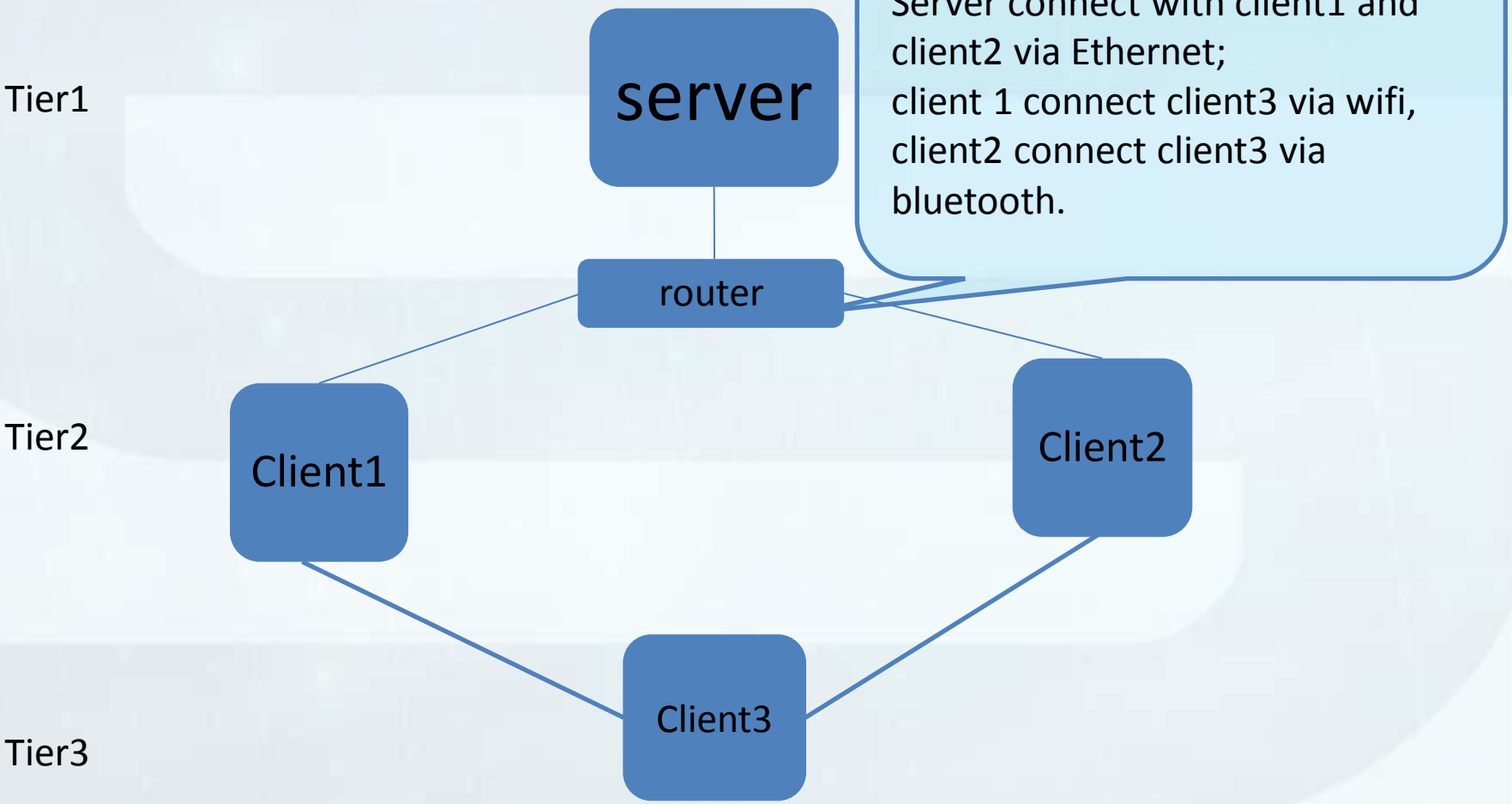
- Initialize server and clients on tree, arrange the Node ID, type and other information to each node.
- New node will broadcast its Node ID and other information via Wifi, bluetooth or enthernet.
- Each time the parent receive the Parent_request. It will judge whether it is a new node, if so, it will send a Parent_confirm.
- New node choose the best parent according to the Tree Based Convergecast Routing Protocol algorithm.

ALGORITHM

- New link has setted up. Data on each node has to be changed
  - eg. One new mobile node connects one tier 2 node. ALL the data on node among this line (tier1, 2 &3) should change.

- The goal of our algorithm is to make system more efficient, which means finding a route to upper tier with least time, least consumption of messages and energy, minimum performance degradation when the number of nodes increases and so on.

TESTBED


Tier1
Tier2
Tier3
server
router
Client1
Client2
Client3
Server connect with client1 and client2 via Ethernet;
client 1 connect client3 via wifi,
client2 connect client3 via bluetooth.

TESTCASE

- Here we list the main testcases.

No.	ACTION	EXPECT RESULT	STATUS
1	Initialization	Server and clients can run successfully	OK
2	Server connect 2 tier-2 clients via Ethernet	Server finds the clients as children while each client connect the server as parent	OK
3	New node (ID:4) conect a tier-2 node (ID:2) via wifi	Tier-2 node (ID:2) connects the new node as child. Node 4 becomes a tier-3 node. Server shows that Node4 is child of Node 2	OK
4	New node (ID:4) conect a tier-2 node (ID:3) via bluetooth	Tier-2 node (ID:3) connects the new node as child. Node 4 becomes a tier-3 node. Server shows that Node4 is child of Node 2	On testing

Evaluation

- In general, we accomplish the goal of this TBCR project
- We can dynamically adjust the whole network(all the nodes) in less than 3 seconds
- The overhead is not high because not all the nodes are involved when there's a change in topology. Only the affected nodes will change their structures accordingly.