Hovering Information: implementation, simulation and analysis

Daniele Bellavista
Scuola di Ingegneria ed Archiettura
Ingegneria Informatica Magistrale
Corso Sistemi Multiagente LM

February 9, 2013

1 Introduction

1.1 Vision

The hovering information is an information dissemination service working in an dynamic infrastructure-free environment with a self-organizing behaviour; a MAS approach may offer a sound paradigm for both hovering information implementation and simulation. The simulation implies the design of a *social* system, where people - hovering information users - move in an environment with different and non-random behaviour. From the simulation results, an analysis of the resulting dynamic network can lead to additional consideration and information that may help understanding and defining service properties and requirements.

In section 2 the system is designed using the *SODA* methodology, in … *TODO*.

1.2 Hovering Information System

Hovering Information is a geo-localized information dissemination service, proposed in [**?**], able to work without a centralized infrastructure. The service is aimed to mobile users capable of peer-to-peer communication and geo-localization. The hovering information system is composed by two main components: mobile nodes and pieces of hovering information.

Mobile nodes are components moving into the environment with a limited communication range, capable of communicate to peers, discover neighbors, access and store (inside a limited buffer) pieces of hovering information. A mobile node is assumed able to determinate its geographic position, speed and direction.

Pieces of hovering information are data that have to *survive* inside a circular area centered at a location called *anchor location* and having a radius called *anchor radius.* The survivability goal of a piece of hovering information is achieved moving or replicating the piece itself through the mobile nodes. A piece of hovering information may have some policies controlling the movement between nodes.

In an hovering information system, three main requirements may be defined for each piece of hovering information [**?**]:

Survivability: a piece of hovering information is alive at some time t, if there is at leas one node hosting a replica of this information.
$survivability = \frac{alive\_time}{total\_time}$

Availability: a piece of hovering information is available at some time t, if there is at least one node in its anchor area hosting a replica of this information.
$avaiability = \frac{avaiable\_time}{total\_time}$

Accessibility: a piece of hovering information is accessible by a node at some time t, if the node is able to get this information; therefore, a replica exists in the node communication range.
$accessibility = \frac{replica\_covered\_area}{anchor\_area}$

2 Hovering Information and Social System analysis and design

The system should implement the hovering information system working inside a social environment. Mobile nodes are owned by people, who move inside an environment composed by anchors, that is locations where pieces of hovering information are present. Anchors are usually bound to points of interest, but in a more general way hovering information can be dynamically created by people.

Mobile nodes lose power and may have not enough energy to supply the whole function. In that case some mobile node features may be limited such as information storage, communication, etc..

The system should simulate an hovering information usage, inside an environment. People, carrying mobile nodes, have to walk with different behaviour, emulating movements inside an area composed by points of interest.

The simulation should gather periodic data about hovering information status and properties (i.e. availability, survivability and accessibility), nodes position and communications link.

2.1 Preliminary analysis

Requirements implicate that the system is composed by three sub-systems:

1. Simulator (graphic interface and data analyzer).
2. Hovering information (composed by pieces of hovering information and mobile nodes).
3. Social system (a group of hovering information users moving into the environment).

Aside from required interfaces, these subsystems can be designed independently from each other and each of them as different users. A simulator user may want to create people and assign a behaviour, nodes and initial hovering information. The simulator itself is the user of the social system and a single person inside the latter system is an user of the hovering information service.

Social system requires people to move with different behaviours; a possible solution is to assign *social roles* with different behavior pattern. Taking the cue from [**?**] some roles can be defined:

Guard: a person who walks following a predefined path.

Employee: a person who resides -works- near a certain point of interest.

Ant: a person who performs a random walk but he's influenced in a certain manner by other people.

Group: some people walking together, using one of the previous behaviour.

2.2 Requirements Analysis

The subsystem *Simulator* doesn't require independent control or intelligence, so the simulator interface can be assumed as a *Legacy System*, already present in the environment.

Requirements Tables

Actor	**Description**
Simulator Starter	The simulator initializer.
Simulation Analyst	The simulation data analyzer.
Person	User of a mobile node.
Mobile node	Hovering information low-level user: discover, access and create the infrastructure for the pieces of hovering information.
Piece of Hovering Information	Hovering information instance.

Table 1: Actor table $(C)Ac_t$

Requirement	**Description**
Access Information	Access all the information that is available in the current position.
Create Information	Create a new hovering information.
Manage Storage	Manage the information stored into the limited buffer of the mobile device.
Analyze Simulation	Analyze the system while the simulation is running.
Walk	Move into the environment.
Survive	Stay alive.
Be available	Be available inside the anchor area.
Maintain accessibility	Cover the maximum possible area inside the anchor area.
Initialize Simulation	Simulator users should be able to specify initial simulation parameters (environment, people, information)

Table 2: Requirement table $(C)Re_t$

Actor	**Requirement**
Person	Access Information, Create Information, Walk.
Simulator Starter	Initialize Simulation.
Simulation Analyst	Analyze Simulation.
Mobile node	Access Information, Create Information, Manage Storage.
Piece of Hovering Information	Survive, Be available, Maintain accessibility.

Table 3: Actor-Requirement table $(C)AR_t$

Domain Tables

External Environment	**Legacy system**
External	Simulator UI.

Table 4: External Environment-Legacy System table $(C)EELS_t$

Legacy System	**Description**
Simulator UI	Simulation output interface that shows simulation data.

Table 5: Legacy System table $(C)LS_t$

Relations Tables

Relation	**Description**
Simulator Data	Make relevant information available to the Simulator UI.
Define Before Simulate	Initial environment definition should occurs before starting simulation.
Create Before Exist	An information has to been created before performing any operation.
Parameters Input	Simulation parameters have to be inserted into the simulation system.
Node Resources	Hovering information should consider node resources when trying to satisfy their requirements.

Table 6: Relation table $(C)Rel_t$

Requirement	**Relation**
Access Information	Simulator Data, Define Before Simulate, Create Before Exist.
Create Information	Simulator Data, Define Before Simulate, Create Before Exist.
Manage Storage	Simulator Data, Define Before Simulate, Node Resources.
Analyze Simulation	Simulator Data, Define Before Simulate.
Walk	Define Before Simulate.
Survive	Node Resources, Define Before Simulate, Create Before Exist.
Be available	Node Resources, Define Before Simulate, Create Before Exist.
Maintain accessibility	Node Resources, Define Before Simulate, Create Before Exist.
Define initial environment	Define Before Simulate, Parameters Input.

Table 7: Requirement-Relation table $(C)RR_t$

Legacy-System	**Relation**
Simulator UI	Simulator Data.

Table 8: Relation-LegacySystem table $(C)RLS_t$

2.3 Analysis

References Tables

Requirement	**Task**
Access Information	access_information
Create Information	create_information
Analyze Simulation	analyze_simulation
Manage Storage	manage_storage
Walk	walk.
Survive	survive.
Be available	be_available.
Maintain accessibility	maintain_accessibility.
Initialize Simulation	define_initial_parameters, start_simulation.

Table 9: Reference Requirement-Task table $(C)RRT_t$

Requirement	**Function**
Access Information	manage_communication manage_node_information
Create Information	manage_communication manage_node_information
Analyze Simulation	inquire_system
Manage Storage	manage_store_buffer
Walk	detect_people.
Survive	manage_communication.
Be available	manage_communication.
Maintain accessibility	manage_communication.
Initialize Simulation	input_initial_parameters.

Table 10: Reference Requirement-Function table $(C)RRF_t$

Requirement	**Topology**
Access Information	Anchor Area, Communication Range
Create Information	Anchor Area, Communication Range
Survive	Anchor Area, Communication Range.
Be available	Anchor Area, Communication Range.
Maintain accessibility	Anchor Area, Communication Range.

Table 11: Reference Requirement-Topology table $(C)RRTo_t$

Requirement	**Dependency**

Table 12: Reference Requirement-Dependency table $(C)RReqD_t$

Legacy System	**Function**
Simulator UI	render.

Table 13: Reference Legacy System-Function table $(C)RLSF_t$

Legacy System	**Topology**

Table 14: Reference Legacy System-Topology table $(C)RLST_t$

Relation	**Dependency**
Simulator Data	SimDataDep
Define Before Simulate	DefBefSimDep.
Create Before Exist	CreateInfDep.
Parameters Input	ParInputDep.
Node Resources	NodeResDep.

Table 15: Reference Relation-Dependency table $(C)RRD_t$

Responsibilities Tables

Task	**Description**
access_information	Access the needed information available in the current position.
manage_storage	Manage the information storage into the limited buffer of the mobile device.
create_information	Create a new hovering information.
analyze_simulation	Analyze information obtained during the simulation.
walk	Move around, basing on behaviour and environment.
survive	Jumps or replicates to mobile nodes in order to keep an high survivability.
be_available	Jumps or replicates to mobile nodes in order to keep an high availability.
maintain_accessibility	Jumps or replicates to mobile nodes in order to keep an high accessibility.
define_initial_parameters	Create initial parameters.
start_simulation	Start the simulation.

Table 16: Task table $(C)T_t$

Function	**Description**
manage_communication	Mobile node p2p communication management.
manage_node_information	Manage information from mobile node.
inquire_system	Get all the information about the system.
render	Show a graphical representation of the simulated system.
manage_store_buffer	Manage storage buffer.
input_initial_parameters	Accept initial simulation parameters as input.

Table 17: Function table $(C)F_t$

2.3.1 Topologies Tables

Topology	**Description**
Anchor Area	Area associated to each hovering information, defined as a circular area with center into the *anchor location* and radius the *anchor radius*.
Communication Range	The maximum effective distance of a *p2p* mobile node communication.

Table 18: Topology table $(C)Top_t$

Task	**Topology**
access_information	Anchor Area, Communication Range.
create_information	Anchor Area, Communication Range.
survive	Anchor Area, Communication Range.
be_available	Anchor Area, Communication Range.
maintain_accessibility	Anchor Area, Communication Range.

Table 19: Task-Topology table $(C)TTop_t$

Function	**Topology**
manage_communication	Communication Range.

Table 20: Function-Topology table $(C)FTop_t$

2.3.2 Dependency Tables

Dependency	**Description**
SimDataDep	access to all information about hovering system components.
DefBefSimDep	define simulation parameters before starting the simulation.
CreateInfDep	information must be created before performing any operation.
SimStartedDep	Simulation should be started.
NodeResDep	Hovering information needs are limited by node resources.

Table 21: Dependency table $(C)D_t$

Task	**Dependency**
access_information	CreateInfDepm, SimStartedDep, SimDataDep.
create_information	CreateInfDep, SimStartedDep, SimDataDep.
manage_storage	NodeResDep, SimDataDep.
analyze_simulation	SimStartedDep, SimDataDep.
start_simulation	DefBefSimDep, SimStartedDep.
walk	SimStartedDep.
survive	SimStartedDep, CreateInfDep, NodeResDep.
be_available	SimStartedDep, CreateInfDep, NodeResDep.
maintain_accessibility	SimStartedDep, CreateInfDep, NodeResDep.
define_initial_parameters	DefBefSimDep.

Table 22: Task-Dependency table $(C)TD_t$

Function	**Dependency**
manage_communication	SimStartedDep, SimDataDep.
manage_node_information	SimStartedDep, CreateInfDep.
inquire_node	SimStartedDep, SimDataDep.
inquire_hovering_information	SimStartedDep, SimDataDep, CreateInfDep.
render	SimStartedDep, SimDataDep.
accept_input	DefBefSimDep.

Table 23: Function-Dependency table $(C)FD_t$

Topology	**Dependency**
Anchor Area	SimDataDep, CreateInfDep.
Communication Range	SimDataDep.

Table 24: Topology-Dependency table $(C)TopD_t$

2.4 Architectural Design

Transition Tables

Role	Task
Simulation Analyst	obtain_nodes_information, obtain_hovering_information, obtain_communication_links.
Person	access_information, create_information, walk
Mobile node	access_information, list_information, create_information, manage_storage.
Piece of Hovering Information	survive, be_available, maintain_accessibility.

Table 25: Transition Role-Task table $(C)TRT_t$

Task	Action

Table 26: Transition Task-Action table $(C)TTA_t$

Resource	Function
MobileCommIf	communicate_data, discover_neighbor.
MobileInputIf	insert_information.
MobileStorageIf	store_information, remove_information.
NodeResource	inquire_node.
HoveringInfResource	inquire_hovering_information.
SimulationInitor	accept_input.
SimulatorUI	render.

Table 27: Transition Resource-Function table $(C)TRF_t$

Function	**Operation**
communicate_data	send, receive.
show_information	show_information.
discover_neighbor	discover_neighbor.
insert_information	insert_information.
inquire_node	inquire_node.
inquire_hovering_information	inquire_hovering_information.
render	render.
store_information	store_information.
remove_information	remove_information.
accept_input	accept_input.

Table 28: Transition Function-Operation table $(C)TFO_t$

Dependency	**Interaction**
SimDataDep	SimDataInt.
DefBefSimDep	DefBefSimInt.
CreateInfDep	CreateInfInt
ParInputDep	ParInputInt.
SimStartedDep	SimStartedInt.
NodeResDep	NodeResInt.

Table 29: Transition Dependency-Interaction table $(C)TDI_t$

Dependency	**Rule**
SimDataDep	SimDataRule.
DefBefSimDep	DefBefSimRule.
CreateInfDep	CreateInfRule
ParInputDep	ParInputRule.
SimStartedDep	SimStartedRule.
NodeResDep	NodeResRule.

Table 30: Transition Dependency-Rule table $(C)TDRu_t$

Topology	**Space**
Anchor Area	Anchor Area Space.
Communication Range	Communication Range Space.

Table 31: Transition Topology-Space table $(C)TTopS_t$

Entities Tables

Action	**Description**

Table 32: Action table $(C)A_t$

Operation	**Description**
send	Send a message to another mobile node.
receive	Receive a message from another mobile node.
show_information	.
discover_neighbor	.
insert_information	.
inquire_node	.
inquire_hovering_information	
render	.
store_information	.
remove_information	.
accept_input	.

Table 33: Operation table $(C)O_t$

Role	**Action**

Table 34: Role-Action table $(C)RA_t$

Resource	**Operation**

Table 35: Resource-Operation table $(C)RO_t$

Interactions Tables

Interaction	Description

Table 36: Interaction table $(C)I_t$

Action	Interaction

Table 37: Action-Interaction table $(C)AcI_t$

Operation	Interaction

Table 38: Operation-Interaction table $(C)OpI_t$

Constraints Tables

Rule	Description

Table 39: Rule table $(C)Ru_t$

Rule	Interaction

Table 40: Rule-Interaction table $(C)IRu_t$

Resource	Rule

Table 41: Resource-Rule table $(C)ReRu_t$

Role	Rule

Table 42: Role-Rule table $(C)RoRu_t$

Space	Rule

Table 43: Space-Rule table $(C)SRu_t$

Topological Tables

Space	Description

Table 44: Space table $(C)S_t$

Space	Connection

Table 45: Space-Connection table $(C)SC_t$

Resource	Space

Table 46: Resource-Space table $(C)ReS_t$

Role	Space

Table 47: Operation table $(C)RoS_t$

2.5 Detailed Design