Hovering Information: implementation, simulation and analysis

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

February 8, 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**
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 reside inside an anchor area.
Create Information	Creates a new hovering information.
Obtain System Data	Known the current data (position, information access, etc.) of all the system component.
Manage Simulation	Manage the simulation.
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.
Define initial parameters	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.
Simulation Analyst	Obtain System Data, Manage Simulation, Define initial parameters.
Mobile node	Access Information, Create Information.
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: show 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 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.

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.
Obtain System Data	Simulator Data, Define Before Simulate.
Manage Simulation	Simulator Data, Define Before Simulate.
Walk	Define Before Simulate.
Survive	Define Before Simulate, Create Before Exist.
Be available	Define Before Simulate, Create Before Exist.
Maintain accessibility	Define Before Simulate, Create Before Exist.
Define initial environment	Define Before Simulate, Create Before Exist, 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	list_information access_information
Create Information	create_information
Obtain System Data	obtain_nodes_information obtain_hovering_information obtain_communication_links
Manage Simulation	start stop pause
Walk	walk .
Survive	survive.
Be available	be_available.
Maintain accessibility	maintain_accessibility.
Define initial parameters	create_information create_people.

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

Requirement	**Function**
Access Information	communicate_data show_information discover_neighbor
Create Information	communicate_data insert_information discover_neighbor
Obtain System Data	inquire_node inquire_hovering_information
Manage Simulation	render
Walk	detect_people.
Survive	replicate jump discover_neighbor.
Be available	replicate jump discover_neighbor.
Maintain accessibility	replicate jump discover_neighbor.
Define initial parameters	accept_input.

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**
Survive	HoverDep.
Be available	HoverDep.
Maintain accessibility	HoverDep.

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

Legacy System	**Function**
Simulator UI	render.
Simulator Creator	accept_input.

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.

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

Responsibilities Tables

Task	**Description**
list_information	List the information available from the current position.
access_information	Access the selected information available in the current position.
create_information	Create a new hovering information.
obtain_nodes_information	Get information of each mobile node of the system.
obtain_hovering_information	Get information of each hovering information of the system.
obtain_communication_links	Get information about current data exchange between mobile nodes.
start	Start the simulation.
stop	Stop the simulation.
pause	Pause 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.
create_people	Create and collocate a new person instance.

Table 16: Task table $(C)T_t$

Function	**Description**
communicate_data	Send data to a mobile node in range.
show_information	Output the requested hovering information data.
discover_neighbor	Find reachable mobile nodes.
insert_information	Input from user data needed for a new hovering information.
inquire_node	Get all the information about a mobile node.
inquire_hovering_information	Get all the information about a piece of hovering information.
render	Show the simulation data.
replicate	Copy an information to another mobile node.
jump	Move an information to another mobile node.
accept_input	Accept initial simulation parameters as user 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**
list_information	Anchor Area, Communication Range.
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**
communicate_data	Communication Range.
discover_neighbor	Communication Range.
insert_information	Communication Range.
replicate	Anchor Area, Communication Range.
jump	Anchor Area, 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.
ParInputDep	Simulation parameters have to be inserted into the simulation system.
SimStartedDep	Simulation should be started.

Table 21: Dependency table $(C)D_t$

Task	**Dependency**
list_information	SimDataDep, SimStartedDep.
access_information	CreateInfDep, SimStartedDep.
create_information	CreateInfDep, SimDataDep, ParInput-Dep.
obtain_nodes_information	SimStartedDep, SimDataDep.
obtain_hovering_information	SimStartedDep, SimDataDep.
obtain_communication_links	SimStartedDep, SimDataDep.
start	DefBefSimDep, SimStartedDep.
stop	SimStartedDep.
pause	SimStartedDep.
walk	SimStartedDep.
survive	SimStartedDep, CreateInfDep.
be_available	SimStartedDep, CreateInfDep.
maintain_accessibility	SimStartedDep, CreateInfDep.
create_people	DefBefSimDep, ParInputDep.

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

Function	**Dependency**
communicate_data	SimStartedDep, SimDataDep.
discover_neighbor	SimStartedDep, SimDataDep.
show_information	SimStartedDep, CreateInfDep.
insert_information	SimStartedDep, SimDataDep.
inquire_node	SimStartedDep, SimDataDep.
inquire_hovering_information	SimStartedDep, SimDataDep, Create-InfDep.
render	SimStartedDep, SimDataDep.
replicate	SimStartedDep, CreateInfDep.
jump	SimStartedDep, CreateInfDep.
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

2.5 Detailed Design