Geo-localized mobile systems: a glimpse of theory

Workshop “Internet of Things”, Oct 21. LAAS-CNRS

Matthieu Roy (TSF / LAAS-CNRS)
roy@laas.fr

Marc-Olivier Killijian, David Powell (LAAS)

François Bonnet (IRISA)
Leodardo Querzoni, Silvia Bonomi (Univ Roma)
Context

* Two fundamental technological shifts:
  * internet -> ambient systems
  * deployment of user-carried systems
  * wireless communication (short range)
  + localization devices
  * link between physical and logical (network) world
Where do we stand?

* Extensive research in “closed” systems
  * abstractions, models, algorithms for resilience

* Extensive research on Internet
  * routing, models, structures (overlays)

* Can we get the “best of both world”
  * i.e. provide localized resilient services
What's the Challenge?

- **Formalization** of the system (local)
- Geographic properties
- **Architectural** design (abstractions or building blocks)
- Inspired by (traditional) distributed ∑
- Development of **algorithms**
- (Assessment on a generic experimental platform)
Is there any application to this?

- Real-life physical examples
- Users deploy a whiteboard
- Perform better GPS route calculation
- Based on users' experience of the traffic
- Cooperative backup of critical data
- Distributed black box, etc.
- Augmented games
## System’s characteristic parameters

<table>
<thead>
<tr>
<th>“classical” systems</th>
<th>mobile systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>failure (node, link)</td>
<td>normal behaviour: disconnections, unreliable wireless communication</td>
</tr>
<tr>
<td>(small) fixed number of nodes</td>
<td>variable and huge size system</td>
</tr>
<tr>
<td>no link between physical world and network</td>
<td>strong coupling with physical environment</td>
</tr>
</tbody>
</table>
So we’ll go local only

- Local = geo-localized
- Everything must be (re)defined w.r.t. a particular location in space.
- Semantics must be coherent with systems’ characteristics:
  - when no user populates a region, it’s not possible to keep a state alive
System definition

* Entities \((p_i)_{i>0}\)
* evolve in 2D space with bounded speed
* equipped with positioning device (infinite precision)
* communication using wireless device
* Let’s concentrate on an area \(A\)
Abstractions for mobile systems

* Traditional distributed abstractions
  * **storage**: registers, transactional memories
  * **agreement**: consensus
  * **group management**: membership

Need to be adapted in geo-aware versions
In this paper, we are interested in providing an abstraction, or building block, that can be directly used by an application. The shared geo-localized storage service, or geo-register, will be built on more simple building blocks.

Even though the systems we consider are large-scale, we focus on an area of interest $A$ for which some basic abstractions are available, as shown in Figure 1. Recall that we suppose that a perfect positioning service is available, and that wireless communication is possible within the whole area $A$. On top of these two basic, hardware-oriented service, we suppose that a geo-reliable broadcast service is implemented. The goal of this service, presented in the next subsection, is to merge geographical and communication-based information to produce a reliable message passing service in the area $A$ with known maximum transfer time.

Our contribution lies in the definition, implementation and proof of the geo-register service, that is presented in the next section, and could be used directly by an application to handle storage in an area $A$.

2.4 Geo-Reliable Broadcast

A geo-reliable broadcast is a communication primitive that guarantees that all processes located in an area $A$ receive the broadcasted message. This primitive is built on top of wireless communication described in the previous section.

More formally the geo-reliable broadcast is defined as follows:

**Definition 4** \((\delta, A)\) geo-reliable broadcast.

Let $\delta$ be a positive number and $A$ be an area. A \((\delta, A)\)-geo-reliable broadcast enjoys the following properties:

- every process $p \in A$ can issue a broadcast $m$;
- if $m$ is a message broadcasted at time $t$ by a correct process $p$ that is in the area $A$ from time $t$ to time $t + \delta$, then all correct processes remaining in $A$ between $t$ and $t + \delta$ deliver $m$ by time $t + \delta$.

This definition is relatively weak, since it does not take into account the processes that may enter or leave the area during the broadcast, and only focuses on entities that stay in the area for the whole duration of a broadcast. The $\delta$ period of time can be either fixed and known in advance or not. In this paper we consider that $\delta$ is fixed for a given $A$, and known by processes.

Such a primitive is implementable when the area $A$ remains valid with the communication graph is a complete graph for the whole period of time $\delta$. 

4 Recall that, in our model, all processes are correct

5 Notice that, in practice, $\delta$ is related to the diameter of $A$. A larger area needs a larger $\delta$ to ensure a reliable dissemination.

Simple example:

**Shared storage/Register**
Geo-reliable broadcast

By hypothesis...

* \((\partial, A)\) geo-reliable broadcast:
  * every process in \(A\) can issue a \(\text{broadcast}(m)\)
  * if \(m\) is broadcasted at time \(t\) by a process that remains in \(A\) from \(t\) to \(t+\partial\) then all processes in \(A\) during \([t, t+\partial]\) deliver the message
Geo-reliable broadcast

But...

* If a process leaves $A$ during the sending interval... no guarantee

* Core region (geographic definition)

* a subset $A'$ of $A$ s.t. every message sent by a process in $A'$ will be delivered by all correct processes that were in $A'$ when the message was sent
Geo-registers

* Simple case: Non concurrent writes
  * write is allowed in the core region $A'$
  * read is allowed in $A$ (roughly)
  * a read operation tries to return the last written value
Non concurrent write semantics

What is the “last written value”? 

* $V = \{ \text{last written value, concurrently written values} \}$ (here $V = \{ y, z, t \}$)

* If, since the last completed write operation,

  * 1) core region was never empty, then $v \in V$ must be returned
  * 2) else it returns $v \in V$ or $\bot$
Geo-registers

Geographically controlled thread:

when \( p \) enters \( A \):
\[
R_p \leftarrow \text{void};
\]
wait for
- \( (W(x)) \) is received : \( R_p \leftarrow x; \) exit;
- \( (2\delta \text{ time delay elapsed}) \)
\text{RB.send}(REQ)

wait for
- \( (REP(v)) \) is received : \( R_p \leftarrow v; \)
- \( (W(x)) \) is received : \( R_p \leftarrow x; \)
- \( (2\delta \text{ time delay elapsed}) \) : \( R_p \leftarrow \bot; \)

when \( p \) leaves \( A \):
\text{free}(R_p);

Communication controlled thread:

\text{upon} \text{ reception of } (REQ) : \text{if } (R_p \neq \text{void}) \text{ then } \text{RB.send}(REP(R_p))

\text{upon} \text{ reception of } (W(x)) : R_p \leftarrow x

Read and Write operations:

When \( p \) is in \( A \):
\text{read()} : \text{wait until } (R_p \neq \text{void}) \text{ then return}(R_p);

When \( p \) is in \( A' \):
\text{write}(x) : \text{RB.send}(W(x));
Properties...

* Region/core region interest:
  * abstracts away physical parameters (network parameters, speed)
  * simple implementation of shared storage

* Semantics:
  * applications that need to store information only when users populate an area
Properties...

* Event-based programming
  * events:
    * application-driven
    * communication/interactions between users
    * movements: interactions with physical world
Extensions

* Current status
  * implementation for one-hop communication model
  * concurrent writers case =? behaviour in presence of failures

* Future work
  * new abstractions/building blocks
Conclusion

* develop building blocks for spatial-based distributed computing
* simple programming
* more resilient applications
* proven building blocks
* new applications?