EMIT

Epidemic Mobile Information Transfer

Author: Jonas M. Kress (Jonas-Kress.de)

Table of Contents

1 Narrative

1.1 The idea

EMIT is a toy that spreads information in an unconventional way and is inspired by nature. More precisely, it tries to adapt models of mechanisms that have emerged in the history of evolution and that have helped to disseminate the most relevant information of the evolutionary process – gene codes.

These models were developed in mathematical biology and help to stop epidemics of infectious diseases. In contrast to biologists we can use these epidemics in a remarkable and valuable way, because we can set the parameters that determine its behavior and also define its payload. As a result, viral information flows can be designed, disseminating information in a surprisingly and adventurous way, which no one could have imagined in detail.

When using mobile peer to peer nodes that are able to build up connections with each other, these flows could stream through channels that are directly established between nearby peers.

As a consequence, a system would be created, which is independent from any given network infrastructure.

Furthermore the profiles of the interacting nodes could be used, when setting the parameters of an epidemic, to gain a more advanced behavior of the viral information dissemination.

1.2 Epidemic model

In the epidemic model a node can be assigned to five stages:

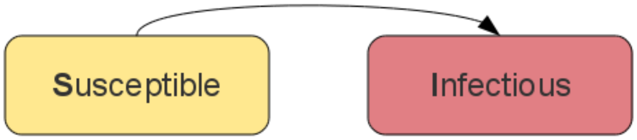
- Susceptible (S)
- Exposed (E)
- Infectious (I)
- Carrier (C)
- Recovered (R)

Usually a peer starts at the stage of being susceptible (S), in the means of being open to receive information, for other peers. After being exposed (E) for a defined period a peer enters the compartment of nodes that are a carrier (C) or that have been infected (I), which means that it is inheriting or spreading the received information.

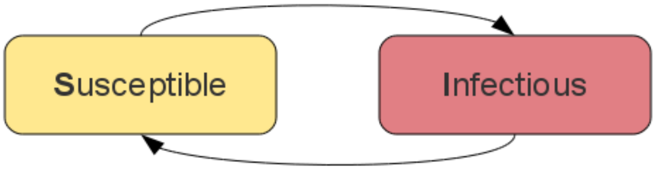
If a node stops disseminating it can be assumed that it has recovered (R) (immune to a specific kind of information) or is susceptible for other peers again.

There are different transition types possible:

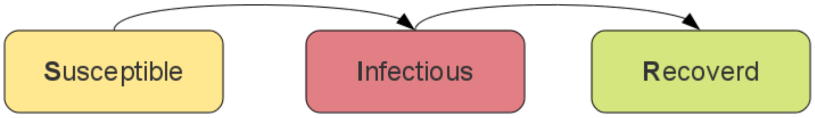
- SI



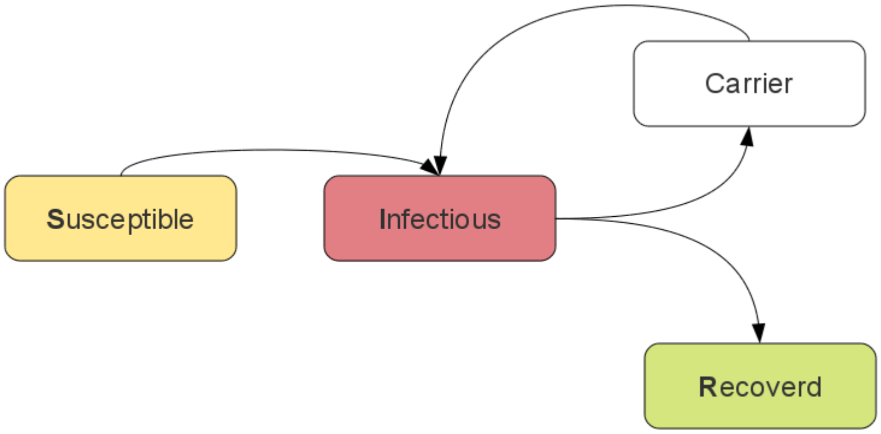
- SIS



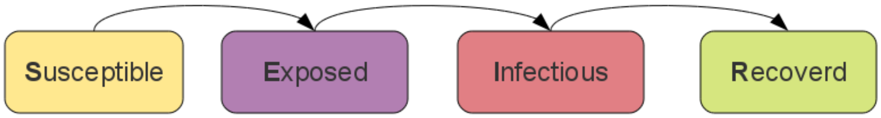
- SIR



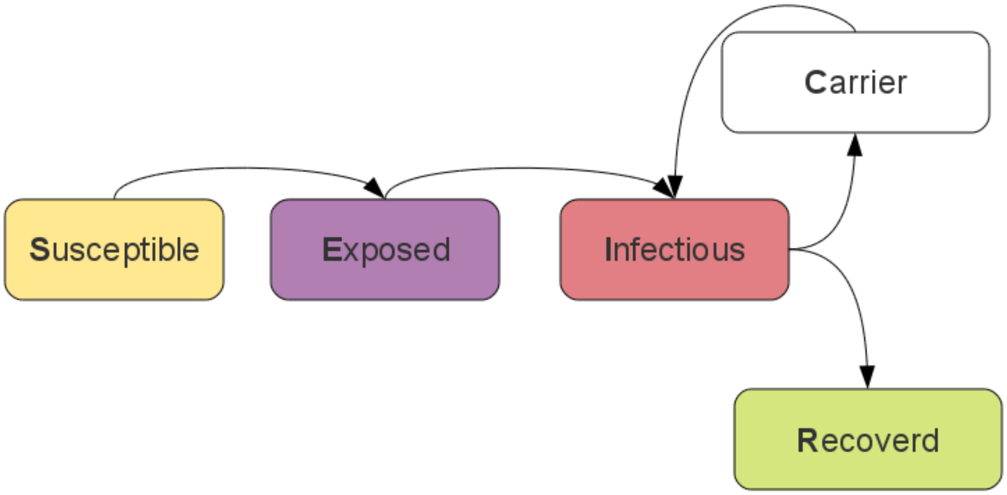
- SIR/C



- SEIR



- SEIR/C



Further information could be read in these papers:

Peter Stechlinski. A study of infectious disease models with switching. Master's thesis, University of Waterloo, 2009

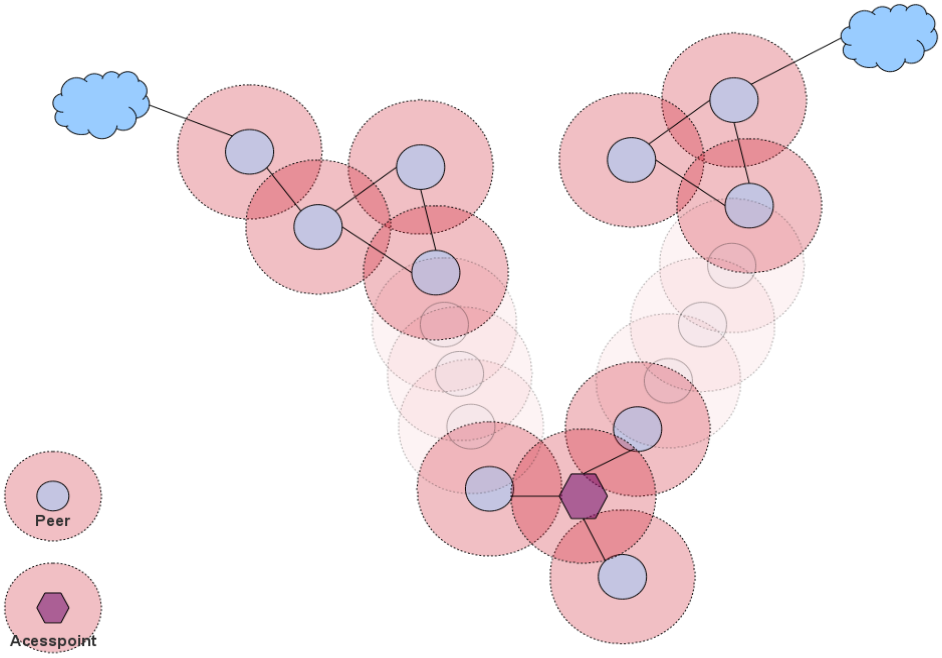
Michalis Faloutsos 3 Nicholas Valler 3 B. Aditya Prakash 1, Deepayan Chakrabarti 2 and Christos Faloutsos 1. Got the flu (or mumps)? Check the eigenvalue! theorem for a wide range of virus propagation models (vpm) that include all virus propagation models in standard literature. Technical report, 1 Computer Science Department, Carnegie Mellon University, Pittsburgh 2 Yahoo! Research, Sunnyvale 3 Department of Computer Science and Engineering, University of California - Riverside, 2010

1.3 Mobile peer-to-peer nodes

Information is spread between mobile peer-to-peer (P2P) nodes. There are various environments in which the mobile peers are able to communicate with each other to disseminate information. Every node is described by a profile representing it's attributes.

1.3.1 Environment

Data is able to spread in diverse environments and scenarios. Through it's WIFI interfaces each node can connect to access-points, which provide a peer-to-peer communication. In the case of an absence of a given network structure peers can establish ad hoc links or build up a hotspot for others. As mobile peers switch to other network partitions and carry information with them opportunistic networks are created. As fail back uplinks such as 2G/3G connections can be used.



1.3.2 Node-Profile

All attributes of each node are represented by the node-profile, which influences the distribution of information implicitly (available interfaces, capabilities and mobility) and deterministic by considering the nodes attributes when spreading an epidemic.

Interfaces

The interfaces of a node are used to communicate with other peers. Mobile devices commonly support the following interfaces:

- 2G/3G
- Bluetooth
- WIFI
- USB
- Camera / Display

Sensors

Sophisticated mobile devices have diverse built in sensors, which are constantly generating measuring data.

Common sensors are:

- Accelerometer (G-Sensor)
- Orientation-Sensor
- Light-Sensor
- Compass
- GPS

Capabilities

As there is a wide variety of hardware capabilities, this data is represented in the nodes profile to gain a wide compatibility. Relevant attributes are:

- Display and camera resolution
- Random access memory
- Available disk space
- CPU power

Mobility profile

The movement of mobile nodes can be detected by analyzing the history of WIFI, 2G/3G and GPS data. This allows the creation of a mobility profiles with these values:

- Movement distances
- Covered routes
- Network partition changes

Social profile

A social profile can represent the interaction of a mobile node owner with his social environment. Possible relevant information:

- Personal data (sex, age, status)
- Friends and networks
- Occupation
- Interests
- Activities

2 Technical

2.1 Peer Components

Every peer consists of four main components:

- Discovery

Implements the functionality to find new nodes.

- Transfer

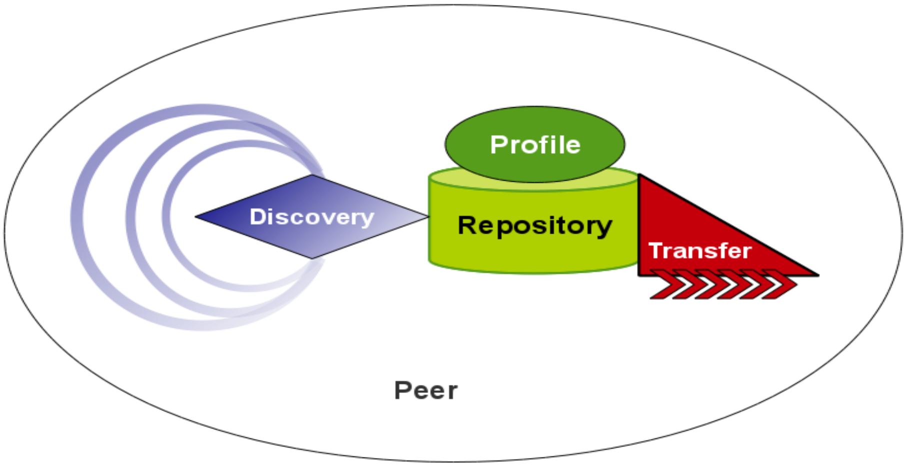
Allows the up- and download of information between two peers.

- Repository

Stores the received information.

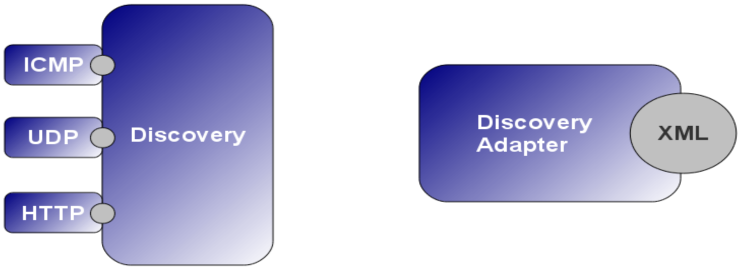
- Profile

Is part of the repository and allows accessing the nodes attributes.



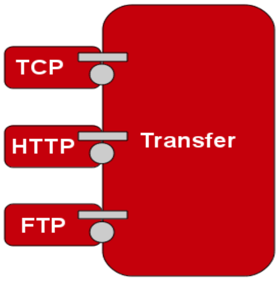
2.1.1 Discovery

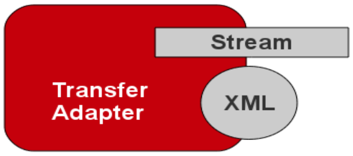
The discovery module uses the nodes network interfaces to scan for other available peers. After finding a foreign peer a node transmits a node-descriptor in the XML format to describe it's own profile and to advertise it's transfer services to other peers. To support multiple network protocols and platform independent implementations adapters are used.



2.1.2 Transfer

The transfer service allows up-/downloading of binary streams and information-descriptors in the XML format. It is announced by the node-descriptors of the discovery module and available under a specified URI. To support generic transfer modes transfer adapters are used.

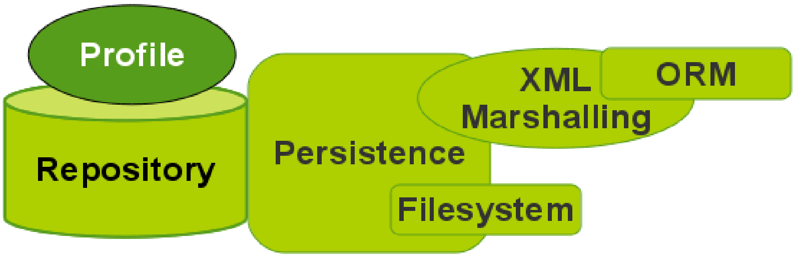




2.1.3 Repository and Profile

The repository uses a persistence component to save and load the received information from the transfer service. The descriptors in XML are transformed into data objects suitable for storing by a marshalling module. An object relational mapper then handles the persistence. Received byte streams are written to the file system by a specific component.

The profile module handles the access to available sensor data and stores a nodes attributes.



2.2 Core

As android supports Java, the three main core components discovery, transfer and repository are implemented in platform independent code. This is achieved by using that parts of the standard Java library that are natively supported by android and sophisticated libraries that allow execution on mobile devices like:

- XStream (http://xstream.codehaus.org)

Java objects to XML serialization.

- ORM-Lite (http://sourceforge.net/projects/ormlite/)

Functionality for persisting Java objects to the SQL databases.

As the profile component is responsible for accessing hardware based sensors it is not part of the core, because it uses platform dependent API's.

2.3 Descriptors

The descriptors are used for inter peer communication of meta data, such as node attributes, information package details and the epidemic behavior. As representation the XML format is used to gain a high interoperability and to be able to use a sophisticated validation of the transferred data.

2.3.1 Node

Every peer uses it's node descriptor to announce the following own information to other peers:

- Transfer services

Data of available transfer services and their URI's.

- Profile

Nodes attributes.

- Certificates

Public keys, key chains and signatures.

2.3.2 Information

As information is transferred between nodes, meta data is needed to describe the parameters of an epidemic, record distribution routes and to attach attributes to a certain package of information. There are various values within an information descriptor, such as:

- Epidemic model (S, I, E, C, R)
  - Delays
  - Profile requirements
  - Events
    - Partition alternation
    - Position change
    - SMS/E-Mail received
    - Sensor measurements
  - Termination
    - Time lived in hops (TL)
    - Time to live in hops (TTL)
    - Time lived as Timestamp
- Routes
  - Points
    - Termination status
    - Position

    - Nodes
- Information
    - Encoding
    - Size
    - Hash
- Signatures

2.4 *Android*

Because Android is based on Linux and supports the Java language as well as parts of the Java library, it allows implementing platform independent code. Of course parts of an Android application are Android specific, because they use special API calls.

2.4.1 Activities and intents

Android uses a sophisticated event handling system with intents and activities. Triggered intents lead to the execution of specified activities. This is used within an application, as well as application comprehensive.

Intent

By starting an activity with an get content action, EMIT is able to allow the user to pick a certain package of information by using other applications. This is done by the following lines of code:

```
intent = new Intent();
intent.setAction(Intent.ACTION_GET_CONTENT);
```

Intent filters

By defining intent filters in the Android manifest file an application can specify that it is interested in handling a class of triggered events. This method allows EMIT to be easy integrated into the sharing system of Android by defining the following lines in the manifest file:

```
<intent-filter>
      <action android:name="android.intent.action.SEND" />
      <category android:name="android.intent.category.DEFAULT" />
      <data android:mimeType="*/*" />
</intent-filter>
```

This code allows EMIT to be executed when a user likes to share a Website, file or Media object.

2.4.2 Service

As Android has a strict lifetime cycle, applications and activities are paused and closed by the system. If a task needs to be executed in the background a service must be implemented. This is done by the following code fragments:

- Extending from an Android Service Class
  ```
  public class EmitService extends Service
  ```

- Defining the Service in the Android manifest file
  ```
  <service android:name=".service.EmitService"></service>
  ```

- Starting the service
  ```
  startService(new Intent(this, EmitService.class));
  ```

2.4.3 Broadcast receiver

To be able to listen to global events like incoming SMS or network location changes a broadcast receiver must be registered. This is done by extending the BroadcastReceiver class and define the receiver in the Android manifest file:

- Broadcast receiver
  ```
  public class EmitReceiver extends BroadcastReceiver {
  ```

- Defining the receiver in the Android manifest file
  ```
  <receiver android:name=".EmitReceiver" android:enabled="true">
  ```

2.4.4 Listener

Accessing the accelerometer data in Android is done by creating a SensorEventListener which is registered to the SensorManager:

- Creating SensorEventListener

```
private final SensorEventListener mSensorListener = new SensorEventListener() {
      @Override
      public void onSensorChanged(SensorEvent se)
```

- Registering sensor listener

```
   mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
   mSensorManager.registerListener(mSensorListener,
    mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
    SensorManager.SENSOR_DELAY_NORMAL);
```

Listening to location change can be done by creating a LocationListener and register it to the LocationManager:

- Creating LocationListener
  ```
  LocationListener locationListener = new LocationListener() {
      public void onLocationChanged(Location location) {
  ```

- Register listener
  ```
  LocationManager locationManager = (LocationManager)
  this.getSystemService(Context.LOCATION_SERVICE);
  ```

```
locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,   0,
0, locationListener);
```

3 Hands-on

There is a great variety of practical scenarios in which the use of EMIT can be really interesting. As the previous explanation were very abstract and theoretically explicit applications can help to achieve a better understanding of EMIT's usefulness.

3.1 Viral Media

We all have experienced trends and media that have been spread over the Internet through blogs and social networks. “We are all susceptible to the pull of viral ideas. Like mass hysteria. Or a tune that gets into your head that you keep on humming all day until you spread it to someone else. Jokes. Urban legends. Crackpot religions. No matter how smart we get, there is always this deep irrational part that makes us potential hosts for self-replicating information.” (Novel Snow Cash by Neal Stephenson). In contrast to usual viral media EMIT is able to spread data in a real viral way, because it is using epidemic models from biology.

3.1.1 Viral TV

We all have seen viral videos that have become very popular, because they were spread by people we know or sites we often visit. Using EMIT viral videos could be used to generate a viral TV stream which is fed by peers surrounding us.

3.1.2 Viral Radio

EMIT enables people to spread their favorite music and allows others to explore new tracks by sending out songs they like.

3.2 Emitting sensors

Emitting sensor data allows swarm measuring, which could help to detect traffic jams, earth quakes or even radioactive pollution.

3.3 Outback messaging

Emit nodes can create their own network infrastructure by connecting to peers that are within reach or by carrying information to distant locations. This could be used in scenarios where common infrastructures are useless, because they are destroyed, overloaded or to expensive to maintain.

3.4 Censorship free communication

Communicating with emit can prevent regimes to censor information, because the data travels through spontaneously established pipes that are established between mobile peers and therefore no infrastructure or central servers are needed.