IoT Notifications: from disruption to benefit

Architectures for the future of notifications in the IoT

**Supervisor(s)**
- Fulvio Corno
- Pino Castrogiovanni

**Presenter**
Teodoro Montanaro
Investigate the intelligence component in Internet of Things (IoT) architectures and applications: study, define, and prototype intelligent distributed architectures that may extract additional value and intelligent behaviors to some significant sample problems, representative of future IoT scenarios.

The distribution and customization of notifications in the IoT domain has been treated as an example of possible future IoT scenarios.
Notification Context: sample scenario

Notification Context: sample scenario

Date: 9th September 2018

Time: 19.00

Date: 9th September 2018

Time: 19.00

5 people:
- Mum: is preparing the washing machine
- Dad: is reading a newspaper
- Clara: is using her pc on her bedroom
- John: is working on his PC
- Frank: is working out on the tapis roulant

Notification Context: sample scenario

Date: 9th September 2018

Time: 19.00

5 people:
- Mum
- Dad
- Clara
- John
- Frank

Notification Context: sample scenario

Date: 9th September 2018

Time: 19.00

5 people:
Mum
Dad
Clara
John
Frank

Various IoT devices

Notification Context: sample scenario

**Date:** 9th September 2018

**Time:** 19.00

**5 people:**
- Mum
- Dad
- Clara
- John
- Frank

**Various IoT devices**

**Events:**
1. Cleaning robot battery is low
2. Frank stops to play sport

Notification Context: sample scenario

**Date:** 9th September 2018

**Time:** 19.00

5 people:
- Mum
- Dad
- Clara
- John
- Frank

**Various IoT devices**

**Events:**
1. Cleaning robot battery is low
2. Frank stops to play sport

Notification Context: sample scenario

Date: 9th September

Time: 19.00

5 people:
Mum
Dad
Clara
John
Frank

Various IoT devices

Going Crazy... Wanna Join?

https://me.me/

Main problem

Notifications could be disruptive:

• **Wrong moment**

• **Wrong device** on which the notification is shown

• **Wrong modality** (e.g., vibration instead of sound)

• **Wrong person(s)**

• **Repetitive** notifications

• Too many **simultaneous** notifications

• ...
Notification Context: sample scenario

Simplified version (used as a reference)

Notification Generator
IoT Sensors / Devices / Services

Cloud Services

Notifications

Notifications

Notified People

Notifications
Main Research GOAL

Design and develop new **IoT architectures** to

a) **enhance** the **effect** of **IoT notifications** on **users experience**

b) allow **developers** to **effectively exploit** the **notifications** improving their services, tools and applications.
Two different approaches are possible

1. At the **distribution level**: notifications are intercepted as soon as they arrive on the IoT devices and then systems decide if, when, and how to show them.
Proposed solutions

Two different approaches are possible

1. At the distribution level: notifications are intercepted as soon as they arrive on the IoT devices and then systems decide if, when, and how to show them.

Solution: **Smart Notification System (SNS)**
Proposed solutions

Two different approaches are possible

2. At the **design level**: notifications are designed with the aim of reducing user disruption.
Two different approaches are possible

2. At the **design level**: notifications are designed with the aim of reducing user disruption.

   Solution: **XDN (Cross Device Notifications) framework**
Smart Notification System (SNS)
**SNS**

**Smart Notification System (SNS):** a modular architecture to deal with notifications at the distribution level.

It uses **machine learning algorithms** to manage incoming **notifications** according to **context awareness** and **users habits**.

**Our contributions:**

1. **Architecture** design
2. **Prototypes** implementation of different architectural components
SNS: Architecture

Overview:

SMART NOTIFICATION SYSTEM

ENVIRONMENT CONTEXT COLLECTORS

ENVIRONMENT CONTEXT ANALYSIS

NOTIFICATION COLLECTOR

Converted Notifications (IN)

USER HABITS

Converted Notifications + LABELS (OUT)

USER CONTEXT COLLECTORS

USER CONTEXT ANALYSIS

DECISION MAKER

DISPATCHER

IoT notifications

Online Services (e.g., Twitter)

IoT notifications

Environment Context information

User Context information

Environment context

User context

Converted Notifications (IN)

Environment Context information
SNS: Architecture

Diagram:
- **Smart Notification System**
  - **Environmental Context Collectors**
  - **User Context Collectors**
  - **Decision Maker**
  - **Notification Collector**
  - **User Habits**
  - **Conversion Analysis**

Connections:
- IoT notifications
- Online Services (e.g., Twitter)
- User Context Information
- Environment Context Information

Processes:
- Environment Context
- Converted Notifications (IN)
- Converted Notifications + Labels (OUT)
- User Context

Outputs:
- Mobile
- Audio
- Light
A modular architecture aware of SNS: Architecture

SMART NOTIFICATION SYSTEM

Environment Context information

ENVIRONMENT CONTEXT COLLECTORS

Environment context

ENVIRONMENT CONTEXT ANALYSIS

Converted Notifications (IN)

NOTIFICATION COLLECTOR

USER HABITS

Converted Notifications + LABELS (OUT)

DECISION MAKER

User context

USER CONTEXT COLLECTORS

USER CONTEXT ANALYSIS

DISPATCHER

User Context information

phone icon, speaker icon, light bulb icon
A modular architecture aware of Environment status (e.g., weather information, current date and time)
A modular architecture aware of User context (e.g., location, status, current activity), Environment status (e.g., weather information, current date and time).
A modular architecture aware of:

- **Environment status** (e.g., weather information, current date and time)
- **User habits** (e.g., usual lunch time)
- **User context** (e.g., location, status, current activity)

**SNS: Architecture**

- **Environment context**
- **User context**
- **Environment context analysis**
- **User context analysis**
- **Decision maker**
- **Converter**
- **Dispatcher**
A modular architecture aware of User habits (e.g., usual lunch time) and Environment status (e.g., weather information, current date and time).

User context (e.g., location, status, current activity).

Decision maker: makes decisions on who should receive the notification, best moment, best devices and best modalities (including actuation) to present notifications.

SNS: Architecture
How can we map our use case?
SNS: Architecture

How can we map our use case?
SNS: Prototypes

1. The **Decision Maker** contribution:
   a) Decision maker prototype

2. The **Collectors** group of contributions:
   a) IoT Collector server
   b) Mobile Collector
   c) SmartHome Collector
   d) SmartCity Collector

3. The **Context Analysis** group of contributions
   a) Location Estimator
SNS: Prototypes

1. The **Decision Maker** contribution:
   a) Decision maker prototype

2. The **Collectors** group of contributions:
   a) IoT Collector server
   b) Mobile Collector
   c) SmartHome Collector
   d) SmartCity Collector

3. The **Context Analysis** group of contributions
   a) Location Estimator
Objective: demonstrate that Machine Learning algorithms can be adopted to the IoT notifications domain
Contribution: Preliminary version of the Decision maker module

Context Information to be used by the ML algorithm:

<table>
<thead>
<tr>
<th>User context</th>
<th>Current activity</th>
<th>Current location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment context</td>
<td>Current timestamp</td>
<td></td>
</tr>
<tr>
<td>Available IoT devices information</td>
<td>Owner</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Current status (e.g., on, off, standby)</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.7 Simplified version of Context information: for the prototype

Notification information to be used by the ML algorithm:

<table>
<thead>
<tr>
<th>Information about incoming notification</th>
<th>Sender</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver</td>
<td></td>
</tr>
<tr>
<td>Type of notification</td>
<td></td>
</tr>
<tr>
<td>Timestamp of receipt</td>
<td></td>
</tr>
<tr>
<td>Assigned labels to outgoing notifications</td>
<td>Target devices</td>
</tr>
</tbody>
</table>

Table 2.8 Simplified version of Notifications information: for the prototype
**Objective**: demonstrate that Machine Learning algorithms can be adopted to the IoT notifications domain

**Contribution**: Preliminary version of the Decision maker module

**Context Information** to be used by the ML algorithm:

<table>
<thead>
<tr>
<th>User context</th>
<th>Current activity</th>
<th>Current location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment context</td>
<td>Current timestamp</td>
<td></td>
</tr>
<tr>
<td>Available IoT devices information</td>
<td>Owner</td>
<td></td>
</tr>
</tbody>
</table>

Current status (e.g., on, off, standby)

Table 2.7 Simplified version of Context information: for the prototype

**Notification information** to be used by the ML algorithm:

<table>
<thead>
<tr>
<th>Information about incoming notification</th>
<th>Sender</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver</td>
<td></td>
</tr>
<tr>
<td>Type of notification</td>
<td></td>
</tr>
<tr>
<td>Timestamp of receipt</td>
<td></td>
</tr>
<tr>
<td>Assigned labels to outgoing notifications</td>
<td>Target devices</td>
</tr>
</tbody>
</table>

Table 2.8 Simplified version of Notifications information: for the prototype
SNS: 1. Decision Maker Prototype

**Objective**: demonstrate that Machine Learning algorithms can be adopted to the IoT notifications domain

**Contribution**: Preliminary version of the Decision maker module

**Tests**:  
- 3 different **machine learning algorithms** adopted over an existing dataset (MIT): Support Vector Machine, Gaussian Naïve Bayes and Decision Trees.

<table>
<thead>
<tr>
<th>ML Algorithm</th>
<th>Percentage of correct predictions with unrelated data</th>
<th>Percentage of correct predictions with related data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy %</td>
<td>Precision %</td>
</tr>
<tr>
<td>Support Vector Machine</td>
<td>81.60</td>
<td>99.89</td>
</tr>
<tr>
<td>Gaussian Naïve Bayes</td>
<td>51.30</td>
<td>99.80</td>
</tr>
<tr>
<td>Decision Trees</td>
<td>99.90</td>
<td>97.06</td>
</tr>
</tbody>
</table>

Table 2.10 Percentage of correct predictions obtained with used algorithms
SNS: 1. Decision Maker Prototype

Objective: demonstrate that Machine Learning algorithms can be adopted to the IoT notifications domain.

Contribution: Preliminary version of the Decision maker module.

Tests:
- 3 different machine learning algorithms adopted over an existing dataset (MIT): Support Vector Machine, Gaussian Naïve Bayes, and Decision Trees.

Main outcome:
- The **three algorithms behave as expected**:
  - **DT works better** than the others due to the programmatic approach used to generate synthetic information.
  - Almost all the algorithms obtain an **high** level of accuracy, precision and recall.
- ML is **promising technique** to enhance the effect of IoT notifications on users experience.

<table>
<thead>
<tr>
<th>Decision Trees</th>
<th>99.90</th>
<th>97.06</th>
<th>99.90</th>
<th>93.90</th>
<th>92.76</th>
<th>93.90</th>
</tr>
</thead>
</table>

Table 2.10 Percentage of correct predictions obtained with used algorithms
SNS: Prototypes

1. The **Decision Maker** contribution:
   a) Decision maker prototype

2. The **Collectors** group of contributions:
   a) IoT Collector server
   b) Mobile Collector
   c) SmartHome Collector
   d) SmartCity Collector

3. The **Context Analysis** group of contributions
   a) Location Estimator
SNS: 2. Collectors

**Aims:**
1. collect real data
2. validate the Machine Learning approach used in the Decision Maker Prototype
SNS: 2.b Mobile Collector

**Aims:**
1. collect real data
2. validate the Machine Learning approach used in the Decision Maker Prototype

**Representative prototype:**
**Objective 1**: collect real user data

- Collect user context information (e.g., location and activity);
- Collect all the mobile and IoT notifications received on user smartphone;
- Collect the user reaction to the received notifications.

**29 people** (5 females and 24 males) used the app for **78 days**
- users receive an average of 247 notifications a day
- users are almost always in the same 3 or 4 places
- users receive most of the notifications from non-important contacts than from important ones.
SNS: 2.b Mobile Collector

Objective 2: validate the Machine Learning approach used in the Decision Maker Prototype

Input features:
- Notification type (mobile, IoT)
- Generating service (e.g., Telegram)
- Ringtone mode
- Notification sender
- Sender-Receiver FAMILY relationship
- Sender-Receiver FRIEND relationship
- Sender-Receiver WORK relationship
- Date and time of receipt (day of week, day of month, month, time)
- User location (Lon/Lat)
- Activity (IN_VEHICLE, ON_BICYCLE, ON_FOOT, RUNNING, STILL, TILTING, UNKNOWN, WALKING)
- Battery level
- Battery status (charging or not charging).
- Connection type (Wifi, network, NoConn)
- Wifi SSID

Label: annoying or appreciated notification
(14 users for 15 days)

Table 2.17 Preliminary results with real data

<table>
<thead>
<tr>
<th>User</th>
<th>% of appreciated notifications over all</th>
<th>Naïve Bayes accuracy</th>
<th>Naïve Bayes precision</th>
<th>Naïve Bayes recall</th>
<th>J48 (Decision Trees) accuracy</th>
<th>J48 (Decision Trees) precision</th>
<th>J48 (Decision Trees) recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>62%</td>
<td>75%</td>
<td>75%</td>
<td>75%</td>
<td>69%</td>
<td>69%</td>
<td>69%</td>
</tr>
<tr>
<td>User 2</td>
<td>70%</td>
<td>94%</td>
<td>94%</td>
<td>94%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td>User 3</td>
<td>96%</td>
<td>92%</td>
<td>91%</td>
<td>92%</td>
<td>95%</td>
<td>91%</td>
<td>95%</td>
</tr>
<tr>
<td>User 4</td>
<td>63%</td>
<td>60%</td>
<td>63%</td>
<td>60%</td>
<td>68%</td>
<td>68%</td>
<td>68%</td>
</tr>
<tr>
<td>User 5</td>
<td>81%</td>
<td>92%</td>
<td>94%</td>
<td>92%</td>
<td>92%</td>
<td>91%</td>
<td>92%</td>
</tr>
<tr>
<td>User 6</td>
<td>65%</td>
<td>58%</td>
<td>56%</td>
<td>58%</td>
<td>58%</td>
<td>53%</td>
<td>58%</td>
</tr>
<tr>
<td>User 7</td>
<td>54%</td>
<td>78%</td>
<td>78%</td>
<td>78%</td>
<td>83%</td>
<td>83%</td>
<td>83%</td>
</tr>
<tr>
<td>User 8</td>
<td>72%</td>
<td>72%</td>
<td>78%</td>
<td>72%</td>
<td>70%</td>
<td>65%</td>
<td>70%</td>
</tr>
<tr>
<td>User 9</td>
<td>57%</td>
<td>67%</td>
<td>68%</td>
<td>67%</td>
<td>60%</td>
<td>59%</td>
<td>60%</td>
</tr>
<tr>
<td>User 10</td>
<td>77%</td>
<td>72%</td>
<td>75%</td>
<td>72%</td>
<td>70%</td>
<td>64%</td>
<td>70%</td>
</tr>
<tr>
<td>User 11</td>
<td>66%</td>
<td>70%</td>
<td>70%</td>
<td>70%</td>
<td>74%</td>
<td>81%</td>
<td>74%</td>
</tr>
<tr>
<td>User 12</td>
<td>61%</td>
<td>82%</td>
<td>82%</td>
<td>82%</td>
<td>91%</td>
<td>92%</td>
<td>91%</td>
</tr>
<tr>
<td>User 13</td>
<td>53%</td>
<td>78%</td>
<td>82%</td>
<td>78%</td>
<td>88%</td>
<td>88%</td>
<td>88%</td>
</tr>
<tr>
<td>User 14</td>
<td>87%</td>
<td>74%</td>
<td>76%</td>
<td>74%</td>
<td>89%</td>
<td>80%</td>
<td>89%</td>
</tr>
</tbody>
</table>

Table 2.17 Preliminary results with real data
SNS: Prototypes

1. The Decision Maker contribution:
   a) Decision maker prototype

2. The Collectors group of contributions:
   a) IoT Collector server
   b) Mobile Collector
   c) SmartHome Collector
   d) SmartCity Collector

3. The Context Analysis group of contributions
   a) Location Estimator
Proposal: demonstrate possibility of inferring user location without energy-hungry methods (e.g., GPS)

People usually spend 85% of their time staying in a few places. The proposed solution uses Decision Trees as Machine Learning supervised classification algorithm to establish user presence in the two most attended meaningful places.

Model that describes the estimation process performed for each user.
SNS: 3. Context Analysis: location estimation

Proposal: demonstrate possibility of inferring user location without energy-hungry methods (e.g., GPS)

Tests:
- 10-fold cross validation through the Weka workbench
- user presence in a meaningful place was estimated every time a new notification is received.
- Input features: combination of Feature Classes (A-AB-ABC-ABCD-ABCDE-BC-...)

Results:
- Most important features (that mainly influence decision) are related to time
- “Current activity” (E) (i.e., the only feature that consumes extra energy), is not necessary
- Accuracy > 75% in almost all tests
XDN Framework
Main Problem: Overwhelming notifications

Second approach

- At the **design level**: notifications are designed with the aim of reducing user disruption
Main Problem: Overwhelming notifications

Second approach
- At the design level: notifications are designed with the aim of reducing user disruption

Developers:
- define their strategies to let their software, then, influence users’ behaviors with respect to notifications
- exploiting the advantages of the cross-device approach
Main Problem: Overwhelming notifications

Second approach

- **At the design level**: notifications are designed with the aim of reducing user disruption

Developers:

- **define their strategies** to let their software, then, influence users’ behaviors with respect to notifications
- **exploiting** the advantages of the cross-device approach
Main Problem: Overwhelming notifications

Second approach
• At the **design level**: notifications are designed with the aim of reducing user disruption
XDN: Our Proposal

XDN (Cross Device Notifications), a framework to assist developers in:

a) **personalizing notifications** to differentiate important and unimportant ones

b) **designing, implementing, and testing cross-device notifications strategies** to inform users without causing too much disruption and involving both mobile and IoT devices.
XDN: Architecture
XDN: Architecture

4 main components:

1. The XDN library
2. The XDN GUI
3. The XDN Runtime Environment
4. The XDN IoT/Mobile library
XDN: Architecture

4 main components:

1. The **XDN library** allows (through APIs) to:
   a) handle incoming notifications
   b) select devices to be involved
   c) perform actions on selected devices
XDN: Architecture

4 main components:

2. The XDN GUI allows developers to explore and evaluate different design alternatives by providing:
   a) an IDE to implement and test developed notification strategies
   b) a simulator to simulate the behavior of the devices
XDN: Architecture

4 main components:

3. The **XDN Runtime Environment** is run on a server to:

- accept device registration requests;
- accept update requests
- accept new notifications
- customize and dispatch the notifications
XDN: Architecture

4 main components:

4. The **XDN IoT/Mobile library** to be integrated in the IoT/mobile applications to:
   - generate notifications compatible with XDN
   - send the generated notifications to the XDN runtime environment;
   - receive commands from the XDN runtime environment (in JSON)
   - execute the received commands.
XDN: Architecture

Excepted behaviour
XDN: Architecture

Excepted behaviour
XDN: Architecture

Excepted behaviour

1. Developer

2. XDN GUI

- Editor
- Simulator
- Simulated Notification sets
- Devices
- Devices portfolio
- Simulated Device sets

XDN runtime environment

IoT/mobile sensors

IoT/mobile devices

XDN IoT/mobile library

XDN API

Dispatcher

XDN library

Developer scripts

Integrate

Distribute notification on
XDN: Architecture

Excepted behaviour

1. Developer
2. XDN GUI
   - Editor
   - Simulator
     - Simulated Notification sets
       - define
       - arrive
     - Simulated runtime environment
       - integrate
       - XDN library
       - API

3. XDN runtime environment
   - IoT/mobile library
   - IoT/mobile sensors
   - Distribute notification on
   - Dispatcher
   - API
   - Devices portfolio
   - Simulated Device sets
   - Devices status
XDN: Architecture

Excepted behaviour

1. Developer
   - XDN GUI
     - Editor
     - Simulator
       - Simulated Notification sets
         - arrive
         - Simulated runtime environment
           - XDN library
             - integrate
             - API
               - Device Status
             - IoT/mobile devices
               - XDN runtime environment
                 - Developer scripts
                   - integrate
                   - API
                   - XDN library
                   - Devices portfolio
                   - Devices status

2. define

3. IoT/mobile sensors
   - XDN IoT/mobile library
     - Device Status
     - XDN runtime environment

4. XDN library
   - Dispatcher
   - XDN IoT/mobile library
     - Distribute notification on
XDN: Architecture

Excepted behaviour

1. Developer
2. XDN GUI
   - Editor
   - Simulator
   - Simulated Notification sets
3. IoT/mobile sensors
4. XDN runtime environment
5. IoT/mobile devices

XDN library

XDN API

Dispatcher
XDN: first prototype

2 components were developed:
1. The XDN library (API)
2. The XDN GUI

Tests with 12 volunteers (11 males and 1 female)

Aims:
• demonstrate the fulfillment of all the requirements
• collect a feedback about APIs and GUI

Each user tasks:
• modify an existing notification strategy
• develop a new notification strategy respecting some given requirements

Volunteers’ main requirement:
• Previous experience with JavaScript
XDN: first prototype

Results:
7 participants over 12 were able to complete all the tasks in the required time.

User feedback: survey (from 0 to 5)

<table>
<thead>
<tr>
<th>XDN Library (API)</th>
<th>XDN GUI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Useful</strong></td>
<td>Simulator</td>
</tr>
<tr>
<td><strong>Complete</strong></td>
<td>Log</td>
</tr>
<tr>
<td><strong>Understandable</strong></td>
<td>Editor</td>
</tr>
</tbody>
</table>

XDN framework in general

- Would be useful the DEBUG function?
- Will you use it in your usual development?
- Would you use it in the future?
- Easy to develop proposed strategies

Table 3.6 - Final survey proposed to user
XDN: first prototype

Results:
7 participants over 12 were able to complete all the tasks in the required time.

XDN Main outcome

- **Efficacy** of the proposed solution to **enhance developers** that want to design, develop and test their own notification strategies.

Table 3.6 - Final survey proposed to user
Thesis Conclusions

Main Problem: Overwhelming notifications

Our proposals:

1. **SNS** that acts at the **distribution** level and fosters **ML algorithms** (autonomous system that directly influences **end-users**)

2. **XDN** that acts at the **design** level and fosters **cross-device approach** (framework for **developers**)

Main outcome:

- **Feasibility** of the proposed approaches was demonstrated
- **Efficacy** of the **proposed solutions** to **enhance**
  - **user experience** with notifications
  - **developers** support in designing, developing and testing their own notification strategies also exploiting the **cross-device approach**
- **Efficacy** of the **user-centered design** methodology in notification domain
Publications during the Ph.D.

2018


2017


• Corno, Fulvio; Montanaro, Teodoro; Migliore, Carmelo; Castrogiovanni, Pino - SmartBike: an IoT Crowd Sensing Platform for Monitoring City Air Pollution. In: INTERNATIONAL JOURNAL OF ELECTRICAL AND COMPUTER ENGINEERING (IJECE, ISSN: 2088-8708, a SCOPUS indexed Journal - Q2), vol. 7 n. 6. (In Press)
Publications during the Ph.D.

2016


2015


• Corno, Fulvio; De Russis, Luigi; Montanaro, Teodoro - **A Context and User Aware Smart Notification System**. In: IEEE 2nd World Forum on Internet of Things (WF-IoT), Milan, Italy, 14-16 December 2015. pp. 645-651

• Montanaro, Teodoro (2015) - **SWARM Joint Open Lab Politecnico Di Torino, Italy**. In: CROSSROADS, vol. 22 n. 2, pp. 70-71. - ISSN 1528-4972
THANK YOU FOR YOUR ATTENTION
Summary of contributions

Main Problem: Overwhelming notifications

Our proposals:

1. **SNS** acts at the **distribution** level and fosters **ML algorithms** (autonomous system that directly influences **end-users**)
   - Decision Maker
   - Collectors
   - Context Analysis

2. **XDN** acts at the **design** level and fosters **cross-device approach** (framework for **developers**)
   - XDN library
   - XDN GUI
   - XDN Runtime Environment
   - XDN IoT/Mobile library