# SMARTIE

SMARTIE is a European project partly founded by the EU. It covers a wide spectrum of areas that include: IoT, M2M, Security policies, ABAC, BigData, NoSQL databases... My work in this area will cover some aspects that could be implemented and used in the context of the SMARTIE project.

http://www.smartie-project.eu/project.html

[**The Vision**](http://www.smartie-project.eu/project.html)

The vision of SMARTIE is to create a distributed framework to share large volumes of heterogeneous information for the use in smart-city applications, enabling end-to-end security and trust in information delivery for decision-making purposes following data owner’s privacy requirements. A secure, trusted, but easy to use IoT system for a Smart City will benefit the various stakeholders of a smart city: The **City Administration** will have it easier to get information from their citizens while protecting their privacy. Furthermore, the services offered will be more reliable if quality and trust of the underlying information is ensured.

Privacy and Trust are a key prerequisite for **citizens** to participate in Smart City activities. A Smart City can improve the Smart and Comfort Live of their citizens enormously.

**Enterprises** benefit from the securely provided information. They can optimize their business processes and deal with peak demands introduced by the dynamics of the Smart City. Furthermore, they can offer more tailored solutions for their customers based on the status of the Smart City.

http://www.smartie-project.eu/project.html

[**Main Objectives**](http://www.smartie-project.eu/project.html)

* Understanding requirements for data and application security and creating a policy-enabled framework supporting data sharing across applications.
* Developing new technologies that establish trust and security in the perception layer and network layer.
* Develop new technologies for trusted information creation and secure storage for the information service layer.
* Develop new technologies for information retrieval and processing guided by access control policies in the application layer.
* Demonstrate the project results in real use cases

http://www.smartie-project.eu/project.html

## [Technical Approach](http://www.smartie-project.eu/project.html)

SMATIE comprises 7 work packages.

WP2 will extract the formal requirements for the IoT platform and it will outline its functional design.

WP3 will investigate and develop novel technologies to securely gather information from the real world e.g. from citizens, traffic control systems etc. and store it in the platform in a privacy-preserving way.

WP4 will work on the development of new techniques to securely retrieve the gathered, processed and stored information from many subsystems, through advanced access control mechanisms, to offer smart and user-tailored services to the citizens. . In addition privacy preserving data manipulation will be researched.

WP5 will be in charge of integrating all the pieces of the puzzle to build the advanced and secure IoT platform to provide enhanced services to the citizens.

WP6 will deploy such framework in a real scenario and evaluate its feasibility.

WP7 will deal with the dissemination of the new knowledge generated within the frame of this project, as well as the exploitation of the achieved results.

http://www.smartie-project.eu/project.html

[**Expected Impact**](http://www.smartie-project.eu/project.html)

* The SMARTIE IoT platform will allow the virtualization of the functionalities of discovery, secure information access, processing and privacy-aware distribution between the consumer and producer of the data generated by the smart objects.
* It will demonstrate the applicability in scenarios linked to the green behaviour and sustainability of smart cities like efficient transport/mobility and energy management.
* SMARTIE will facilitate new companies for developing and providing services over its IoT infrastructure/platform.
* SMARTIE allows heterogeneous and multiple source of data to interact in reliable and secure manner providing third parties developers in Europe to enter the Smart Cities area and in that way increase the share of the IoT market.

http://www.smartie-project.eu/project.html

[**Use Cases**](http://www.smartie-project.eu/project.html)

**Frankfurt/Oder (GER)**

* Traffic management with the possibility to influence real traffic.
* Focus on authentication, trust, data security, interoperability

**Murcia (ES)**

* Monitoring energy efficiency in the campus
* Users can interact with the system to improve energy efficiency.

**Belgrade (RS)**

* Provide smart transportation using location of busses and travellers
* Focus on data security and privacy using developed access rights and policies

SMARTIE USE CASES

Traffic, Energy, Security, Information, IoT

Work packages 3, 4 and 5 focus on designing and developing required components that will be able to

address these issues. In particular, WP3 is focused on development of novel technologies to securely gather

information from the citizens and store it in the platform in a privacy-preserving way. More specifically, the

objectives are to:

 provide light-weight and scalable security and trust mechanisms for the constrained IoT nodes

 design IoT node attestation concepts and approaches

 provide mechanisms that enable integrity and confidentiality in information gathering

 define secure data transport requirements between IoT devices and cloud infrastructure

 provide concepts and approaches related to the cryptographic data storage

# RBAC-ABAC

# NoSQL

A comparison between several NoSQL databases with comments and notes

Bogdan George Tudorica, Cristian Bucur

Department for Economical Mathematics and Economical Informatics

Petroleum-Gas University of Ploiesti

Ploiesti, Romania

tudorica\_bogdan@yahoo.com

The concept described by the term NoSQL (meaning a database system which is distributed, may not require fixed table schemas, usually avoids join operations, typically scales horizontally, does not expose a SQL interface and may be open source [1] – some are even using the term with the meaning of a completely non relational system) is also referred by the more academic sources as a form of structured storage [4][10][11][12] (although the terms may not be equivalent; the relational databases also comply by the official definition of the structured storage term and they are somehow opposite to the NoSQL term).

One can not simply label the terms RDBMS and NoSQL as being the exact opposite. There do even exist some middleware appliances (such as CloudTPS for Google’s BigTable and Amazon’s SimpleDB [17]) or various solutions (such as Percolator for Google’s BigTable [14] and an unnamed prototype system for Google’s Hbase [7]) which are adding full ACID features to some NoSQL systems.

It is certain that the NoSQL databases are one of the byproducts of the Web 2.0 era – they were really used only at the time when the designers of web services with very large number of users discovered that the traditional relational database management systems (RDBMS) are fit either for small but frequent read/write transactions or for large batch transactions with rare write accesses, and not for heavy read/write workloads (which is often the case for these large scale web services – we mean Google, Amazon, Facebook, Yahoo and such).

In order to be able to compare a set of NoSQL solutions the first step should be to select / classify some products which are fulfilling similar purposes or have similar qualities / features.

For the moment there is no official taxonomy for this kind of software although several attempts do exist.

First one is provided by Stefan Edlich on his page [8] and it is providing the following categories:

A. Core NoSQL Systems, most of them created as component systems for Web 2.0 services, with the following subtypes:

• Wide Column Store / Column Families (Hadoop / HBase, Cassandra, Hypertable, Cloudata, Amazon SimpleDB, SciDB),

• Document Store (CouchDB, MongoDB, Terrastore, ThruDB, OrientDB, RavenDB, Citrusleaf, SisoDB, CloudKit, Perservere, Jackrabbit),

• Key Value / Tuple Store (Azure Table Storage, MEMBASE, Riak, Redis, Chordless, GenieDB, Scalaris, Tokyo Cabinet / Tyrant, GT.M, Keyspace, Berkeley DB, MemcacheDB, HamsterDB, Faircom C-Tree, Mnesia, LightCloud, Pincaster, Hibari, Scality),

• Eventually Consistent Key Value Store (Amazon Dynamo, Voldemort, Dynomite, KAI, SubRecord, Mo8onDb, Dovetaildb),

Graph Databases (Neo4J, Infinite Graph, Sones, InfoGrid, HyperGraphDB, Trinity, AllegroGraph, Bigdata, DEX, OpenLink Virtuoso, VertexDB, FlockDB, Java Universal Network / Graph Framework, Sesame, Filament, OWLim, NetworkX, iGraph),

December 2013

An Access Control Model for NoSQL Databases

Motahera Shermin

*The University of Western Ontario*

Supervisor

Dr. Sylvia Osborn

*The University of Western Ontario*

The main promoters of NoSQL databases are Web 2.0 companies such as Amazon,

Twitter, LinkedIn and Google which are challenged with unprecedented data, operation

volumes under tight latency constraints and massive infrastructure needs. NoSQL

databases represent the much needed data storage evolution in enterprise application

architecture, continuing the evolution of the past twenty years. In the 1990’s, vertically

integrated applications gave way to client-server architecture, and with Web 2.0, clientserver

architecture transcended to three-tier web application architecture. In parallel, the

demands of web-scale data analysis added map-reduce processing into the mix and data

scientists started trading transactional consistency for incremental scalability and largescale

distribution. The NoSQL movement emerged out of the later constellation [34].