*SBC Reviews on Computer Science, 2021, 1:1,* doi: 10.5753/reviews.2021.1805


# Decentralized, distributed and fault-tolerant context recognition architectures for smart cities: A systematic mapping

**Leonardo Vianna do Nascimento** [**Instituto Federal do Rio Grande do Sul** | ***leonardo.nascimento@alvorada.ifrs.edu.br***]
**José Palazzo Moreira de Oliveira** [**Universidade Federal do Rio Grande do Sul** | ***palazzo@inf.ufrgs.br***]

**Abstract**
An essential aspect of smart city applications is how to obtain context data about their users and understand them. Several works in the last decades have presented context recognition software architectures that can be applied in smart cities. This study aims to overview the state of the art in distributed context recognition software architectures suitable for smart cities and identify research opportunities. The researchers conducted a systematic mapping that provides an overview of such architectures applied in the smart cities domain. This review focuses on the following aspects of these approaches: decentralization, plugin support, resilience, data fusion, dynamic composition, privacy, and security. The researchers searched for relevant papers using four search engines (ACM Digital Library, IEEE Xplore, Scopus, and Springer Link). The papers returned by this search passed by a two-step filtering process: the analysis of title, abstract, and keywords; and an analysis of sections that describe the proposed solution. The selected papers include 87 works of the total 1977 papers returned in the search. The analysis of these papers has shown that only a few works explored resilience, data fusion, security, and privacy, and fewer integrated these aspects in decentralized architecture with plugin support and dynamic composition. The development of architectures that integrate these aspects is a research opportunity, especially those that present resilience and security-related mechanisms.

***Keywords:*** *Context-aware, Architectures, Decentralized, Fault-Tolerant, Data Fusion, Systematic Mapping*

## 1 Introduction

According to a recent United Nations report, most of the world's population live in cities (United Nations, Department of Economic and Social Affairs - Population Division, 2018). This large concentration of people in cities raises several challenges related to the economy, mobility, sustainability, security, governance, and quality of life. Therefore, cities should be more intelligent to improve the quality of the services offered to their citizens (Nam and Pardo, 2011).

A *smart city* must integrate technologies, systems, services, and capabilities into an organic network (Albino et al., 2015).In this context, the Information and Communication Technologies (ICTs) are a crucial element that can provide the necessary infrastructure to make this integration possible. The work of Talari et al. (2017) cites several applications using ICT in smart cities, such as smart parking, smart health, surveillance systems, traffic management, water systems, and environmental pollution monitoring.

These applications may acquire and apply contextual data to improve the experience of users (Morandi et al., 2016). Context-aware applications can receive data from sensors, perform inferences on these data to obtain higher-level context information, and use them somehow. Sensors are a valuable source of data about resources, places, services, users, and contexts. Recently, there has been a remarkable growth of digital devices, such as sensors, actuators, smartphones, and smart appliances in cities (Rathore et al., 2016).

Because of the large number of commonly present sensors in smart cities, the traditional approaches that connect applications directly to sensors become infeasible. Context-aware specific software architectures are needed to manage sensor data, generate high-level context data, and distribute them to applications (Perera et al., 2013).

These architectures should make it possible that application maintainers can add new sensors as efficiently as possible. The information provided by these sensors can be fused and analyzed to provide high-level context information.

Traditional centralized context-aware architectures applied to process the high amount of data provided by sensors in smart cities can lead to performance bottlenecks. Cloud computing technologies can provide high-performance computational resources that can fastly process this data (Al Nuaimi et al., 2015). Unfortunately, cloud computing services processing this amount of data can lead to high latency and high network traffic. Therefore, distributed and decentralized architectures are desirable for context recognition in smart cities. These architectures can use paradigms like *fog computing* and/or *edge computing* to decentralize and distribute data processing on several physical nodes(Sánchez-Corcuera et al., 2019).

Furthermore, cities are very dynamic and open environments. Applications in a smart city domain do not have to deal with a controlled environment. Unexpected events can lead to failures in sensors, communication links, servers, and other devices. Components may depend on or interact with each other in sophisticated ways, and transient errors that previously occurred in isolation can now quickly cascade into failures throughout the whole intelligent system. Therefore, context-aware architectures in smart cities should present some degree of fault tolerance to mitigate the increasing potential for failure in their applications (Alegre et al., 2016).

There are several context recognition approaches presented in the literature (Nascimento et al., 2021). However, the work of Perera et al. (2013) highlights that each technique has its strengths and weakness and that no one alone can accomplish perfect results. Therefore, the best method to tackle the problem of context recognition is to combine

these models in such a way that they can complement each other. Thus, an architecture should easily aggregate new context recognition models. Furthermore, the selection of these models could be based on the situation and application needs. This kind of capacity is called *dynamic composition*.

Security and privacy are important issues related to distributed context recognition systems. Malicious users can intercept user data sent through network connections. Therefore, distributed context recognition architectures should provide mechanisms to guarantee the privacy and ensure the security of transactions Perera et al. (2013).

Thus, decentralization, resilience, capacity to add new sensors and context recognition models as plugins, fusion mechanisms, and support to privacy and security are desirable requisites of context recognition architectures. This work systematically maps the computer science literature to investigate distributed context recognition architectures applied in smart cities and analyze the application of the previously stated requisites. This review is justified as it allows smart city researchers and systems developers to learn how the previous five concepts have helped develop context recognition architectures in the past. It also helps to understand how we can apply those techniques to develop new solutions that efficiently take advantage of smart city infrastructures. Finally, the identified gaps present future research directions.

The main contributions of this paper are:

- an overview of distributed software architectures for user context recognition in smart cities focused on the analysis of the existence of decentralization, resilience, plugin support, privacy, security, and fusion mechanisms;
- the identification of challenges, open issues, and the possibilities of future works related to context recognition software architectures in smart cities.

The rest of this paper is organized as follows. Section 2 shows some concepts about smart cities and context recognition architectures. Section 3 presents and discuss related works. Section 4 shows the systematic mapping study methodology. Section 5 presents the analysis, categorization, and discussion of the selected papers. Finally, Section 7 presents some conclusions and future works.

# 2 Background

## 2.1 Smart Cities

There are many definitions of smart cities. Each one emphasise different aspects of what is a smart city. Several of them are listed in Albino et al. (Albino et al., 2015) and analyzed in the work of Fernandez-Anez (Fernandez-Anez, 2016). The majority of these definitions are centered on people and that a smart city must improve the citizens' quality of life. The economy is an important aspect, and a smart city should provide sustainable development. There are several smart city initiatives around the world with different maturity levels and different application domains. Most of them are in Europe[1], North America[2], and Asia[3].

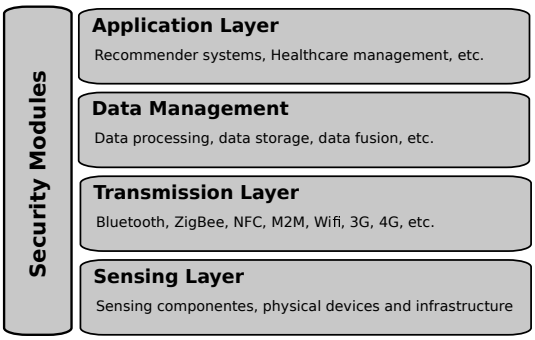


**Figure 1.** Generic architecture of a smart city (Silva et al., 2018)

Technology, especially ICT, is a fundamental component to provide the necessary infrastructure to improve the cities' services. Sánchez-Corcuera et al. (2019) identify that the most common architectures applied in these solutions are *cloud computing*, *fog computing* and *edge computing*. These architectures are complementary and can be implemented in the same city to reduce network traffic and obtain better-balanced computational resource usage. Some other fundamental technologies used in smart cities solutions have been highlighted by Khatoun and Zeadally (2016): *big data*, *IoT*, *service-oriented architectures* and *cybersecurity architectures*. IoT provides essential building components for smart cities, mainly in data generation and transmission. Big data approaches are often applied in managing huge amounts of data generated by the IoT infrastructure. Service-oriented interfaces (e.g., Web services) can be used by applications to obtain data from lower layers.

A generic architecture for smart cities has been presented in Silva et al. (2018). The proposed architecture comprises four layers (Figure 1).

Data collection is the primary responsibility of the sensing layer. Physical sensors are common sources of data. Smart cities' solutions widely use physical sensors deployed in IoT infrastructures (Talari et al., 2017). Smartphone sensors are a widespread source of data as well (Nascimento et al., 2021). Other data sources in smart cities include Web services, social networks, crowdsensing, and open data (Kon and Santana, 2016; Khatoun and Zeadally, 2016).

The obtained data are carried to the upper layers by the transmission layer via various communication technologies. There are several platforms that provide data collection and transmission capabilities, such as FIWARE (Cirillo et al., 2019) and KNoT (Batista et al., 2018).

The data management layer processes and stores valuable information for service provision offered by multiple applications at the top layer. Context recognition approaches are applied in this layer to generate context information and make it available for applications. Data protection is a crucial concern of any smart city. Thus security modules should be integrated into each layer.

[1] http://www.smart-cities.eu

[2] http://www.fastcoexist.com/3021592/the-10-smartestcities-in-north-america

[3] https://hub.beesmart.city/en/strategy/en/rising-asian-smart-cities-to-watch-in-2019-parttwo#:~:text=When%20people%20think%20of%20Asian,enabled%20metropolises%20in%20the%20region

## 2.2 Context Recognition

There are several definitions for context in computer science literature (Bazire and Brézillon, 2005). In this work, we used the definition of Dey & Abowd, where *context* is *"any information useful to characterize the situation of an entity (a person, object, or place) that can affect the interaction between users and systems"* (Abowd et al., 1999).

(Attard et al., 2013) categorize context in three levels: *raw*, *interpreted* and *situation*. Raw context is data provided directly by sensors and other data sources. Interpreted context information is derived from raw data. It represents the information in a higher level of abstraction (e.g., activities such as running or working, place types such as home or food establishment). Situations are combinations of interpreted context patterns that can be viewed as static states associated with an entity, and any state change is another situation (Gundersen, 2013). Thus, examples of situations include *working*, *driving to home*, *in a meeting*.

The context management process consists of four phases (Perera et al., 2013) : *context acquisition* (where the context data came from), *context modelling* (how context data are represented), *context recognition* (how context data are processed to generate high level information) and *context dissemination* (how context information are distributed to the consumers). A context recognition architecture must include components to perform activities related to all of these phases.

Different data sources execute context acquisition. As stated in Section 2.1, common raw context data sources in smart cities are IoT sensors, smartphone sensors, Web services, and social networks.

The context recognition process obtains high-level context from low-level context. Computer science literature proposed different context reasoning mechanismsPerera et al. (2013) based on machine learning techniques such as *supervised learning* (decision trees, Bayesian networks, support vector machines, artificial neural networks), *unsupervised learning* (clustering techniques, unsupervised neural networks), *rules*, *fuzzy logic*, *ontological reasoning* and *probabilistic reasoning* (Dempster-Shafer, hidden Markov Models). Details about the usage of these mechanisms are not the focus of this work but the organization of the different reasoning components in a software architecture.

Perera et al. (2013) also presents the most popular context modelling techniques: *key-value pairs*, *markup schemes*, *graphical*, *object based*, *logic based*, and *ontology based modelling*. . Some proposals use hybrid context representations, applying different techniques according to levels (Machado et al., 2018).

Context dissemination provides methods to deliver context to the consumers. There two common methods used in context distribution (Perera et al., 2013): *query* and *subscription* (also called *publish/subscribe*). A consumer requests a query, and the context manager can use it to produce a result. In a subscribe method, a consumer specifies a set of requirements and can be notified periodically or when an event occurs.

The focus of this work is the description of user context recognition architectures. The user context concept used in this paper is related to the information exclusively related to the user, such as user location or user activity in an urban environment.

## 2.3 Software Architectures

This work focus on analyzing the software architectures of context recognition subsystems in smart city solutions. Bass et al. (2012) defines a *software architecture* as "the set of structures needed to reason about the system, which comprises software elements, relations among them, and properties of both."

An important aspect to consider when analyzing software architectures is their quality attributes. A quality attribute is a measurable or testable property of a system used to indicate how well the system satisfies the needs of its stakeholders (Bass et al., 2012). Our analysis of context recognition software architecture covers the following quality attributes identified in the literature (Bass et al., 2012):

- *Availability* is a property of software related to its capacity to be ready to carry out its tasks when needed. This attribute encompasses reliability and adds the notion of recovery or resilience (the capacity of a software module to repair itself after a failure). We analyzed the resilience and fault-tolerance mechanisms present in context recognition architectures in Section 5.7.
- *Interoperability* is about the degree to which two or more systems can usefully exchange meaningful information via interfaces. Context recognition architectures are part of more extensive context-aware systems that consume the context information provided by them. Thus, the interfaces provided by context recognition subsystems are a crucial aspect related to their interoperability capacity. These interfaces are related to the context distribution mechanisms discussed in Section 2.2. We present an analysis of context distribution approaches used by context recognition architectures in Section 5.5 as well as how ontologies contribute to improving the semantic interoperability of these interfaces.
- *Modifiability* is related to the cost and risk of changes in software systems. It is a crucial aspect of context recognition architectures in smart cities as these environments are very dynamic. These systems should adapt to changes in the environment, such as new sensing capabilities and the necessity of new reasoning mechanisms to deal with these data. We present the modifiability aspect of selected context recognition architectures in Section 5.7 when analyzing the plugin support of these solutions.
- *Performance* is about the ability of a software system to meet timing requirements. Data derived from the urban data sources is characterized by heterogeneity, and it typically is of big data scale (Moustaka et al., 2018). Distributed reasoning mechanisms are frequently used to process this high volume of data and achieve timing requirements(Al Nuaimi et al., 2015). This work selected only distributed context recognition architectures. We consider both architectures that deal with distributed

data and architectures that use distributed processing. However, the processing time is not the only aspect related to performance in context recognition solutions in smart cities. Latency is another crucial aspect of these systems and is related to the latency of network communication infrastructures. Decentralized architectures can reduce latency by providing processing nodes near devices that acquire data (Sánchez-Corcuera et al., 2019). Section 5.7 discusses the application of decentralized architectures in the selected context recognition solutions.

- *Security* is the ability of a software system to protect data and information from unauthorized access. We discuss in Section 5.6 the security and privacy mechanisms provided by distributed context recognition architectures in smart cities.
- *Safety* is about the software's ability to avoid entering states that cause or lead to damage, injury, or loss of life to actors in the software's environment. Context recognition solutions are not used directly by users and do not act on the system. They are responsible for getting data from their environments and reason about these data. However, consumer applications can use context data to decide what actions to perform. In this situation, context recognition subsystems in smart cities fuse sensor data, combining sensor data from multiple sensors to produce more accurate, more complete, and more dependable information that could not be achieved through a single sensor (Perera et al., 2013). We analyze and discuss sensor data fusion support in the selected context recognition architectures in Section 5.7.

The study presented in this work does not cover quality attributes related to the usage of context data by applications such as usability or ethics. Context recognition subsystems are not used directly by users, and such quality attributes do not apply to them.

Other important aspect related to software architectures are *patterns*. Architectural patterns are ways of capturing proven good design structures, so that they can be reused (Bass et al., 2012). We considered in these work 11 known architectural types listed in Bass et al. (2012): *multi-tier*, *layered*, *service-oriented*, *broker*, *client-server*, *peer-to-peer*, *publish/subscribe*, *shared-data*, *pipe-and-filter*, *micro-kernel*, and *map-reduce*. We also considered the multi-agent paradigm as a pattern, as several papers presented a multi-agent approach to obtain and process context data.

# 3 Related Works

Recent surveys, literature reviews, and systematic mappings analyzed context-aware research in the literature applied for different domains. This section summarizes related works.

Perera et al. (2013) presents a survey that analyzed 50 context-aware projects conducted between 2001 and 2011. The work used an evaluation framework composed of 20 items, including modeling, reasoning, distribution techniques, architectural pattern, dynamic composition, security, and privacy analysis.

Li et al. (2015) presents a survey of context-aware middleware architectures from 2009 through 2015. Eleven selected papers were analyzed and compared based on technical parameters, including architectural pattern, context reasoning, scalability, fault tolerance, context awareness level, security, and privacy. The analysis shows that there is no context-aware middleware architecture that complies with all requirements.

The work of Roda et al. (2018) presents a systematic mapping that aims to analyze the existing context-aware architectures in different domains. A process of search and filter selected 463 papers published between 1999 and 2017. Smart environments and healthcare are the domains of application of the identified architectures. The paper identified that the most common architecture concepts are *multilayered*, *component-based*, *service-oriented*, and *multiagent*. Multilayered architectures are common because they provide a simple form to separate concerns, where each layer should provide different functionality. Such functionalities are commonly implemented by combinations of other types, such as multiagent systems services and components.

All works present further analysis of architecture patterns, and two of them offer an analysis of security and privacy. One paper presents an analysis of fault tolerance in context-aware architectures, and just one another presents an analysis of dynamic composition in these architectures. Therefore, there are no works that overview distributed context recognition architectures in smart cities, highlighting decentralization, resilience/fault-tolerance, dynamic composition, fusion mechanisms, and sensor management.

# 4 Systematic Mapping Study

A *systematic mapping study* is an approach designed to provide a comprehensive overview of a research area, where a piece of evidence in a domain is plotted at a high level of granularity Kitchenham et al. (2007). It allows the identification of research trends and areas for future studies. The study presented in this paper followed the steps that were proposed by Petersen et al. Petersen et al. (2015): (i) definition of research questions; (ii) searching for relevant studies; (iii) study selection; and (iv) data extraction and mapping.

## 4.1 Research Objectives

This systematic mapping study aims to investigate distributed software architectures to recognize user context related to the smart city domain. We focus on identifying software architectures that present a capacity to include new sensors/inference mechanisms through plugins and apply the concepts of decentralization, resilience/fault-tolerance, dynamic composition, and context information fusion. We also analyzed other characteristics of such solutions: architectural patterns, data source usage, security & privacy support, context modeling techniques, context delivery approaches, validation methods, and research opportunities.

## 4.2 Research Questions

Aiming to support the research objective, we defined eight Research Questions (RQ) presented below.

- **RQ1**:*What is the distribution of works per year?*
- **RQ2**:*Which are the most evident publication venues?*
- **RQ3**:*Which architectural patterns are used?*
- **RQ4**:*How context information is managed (acquisition, modeling and distribution)?*
- **RQ5**:*How do context recognition architectures deal with security/privacy issues?*
- **RQ6**:*How do context recognition architectures apply the concepts of decentralization, resilience, dynamic composition, context information fusion, and plugin support?*
- **RQ7**:*What validation methods do the works often employ?*
- **RQ8**:*What are the research opportunities?*

### 4.3 Search Strategy

Aiming to perform the search to answer our RQs, we defined the following search string: *(architecture OR design OR framework OR component OR infrastructure OR specification OR middleware) AND ("context-aware" OR "situation-aware" OR "context-sensitive" OR "environment-directed") AND (user OR human) AND ("city" OR "smart environment" OR "urban" OR ubiquit* OR pervasive )*. It includes terms from the research objective, which is the primary purpose of this mapping, words from the research questions, and synonyms.

We performed searches for primary studies using the following databases and search tools: *ACM Digital Library*, *IEEE Xplore*, *SpringerLink*, and *Scopus*. The search string has been applied in the titles and keywords of the articles presented in the databases. Unfortunately, SpringerLink does not have an option to restrict the search in title and keywords. Because of this, the search performed on it was less restrictive.

### 4.4 Selection Criteria

To filter studies, we applied some inclusion and exclusion criteria, enabling us to answer our RQs. We employed the following inclusion (I) and exclusion (E) criteria:

- **I1.** The study is a scientific paper from conferences, workshops, or journals related to Computer Science.
- **I2.** The study is published before 2020.
- **E1.** The study is not written in English.
- **E2.** The study is a short one, less than four pages.
- **E3.** The paper does not present a primary study.
- **E4.** The study is duplicated.
- **E5.** The study is an older publication from the same authors about the same approach.
- **E6.** The study does not present an architecture for a user context-recognition system with a distributed context recognition approach or acquire data from a distributed sensor arrangement.
- **E7.** The full text of the primary study is not available.
- **E8.** The study does not present an approach that is not domain-specific or applied to the smart cities domain.

### 4.5 Selection of Studies

We conducted the systematic mapping study from July to December 2020. The search string was applied to the four search tools and obtained 1977 papers[4]. During this phase, we carried out two revision rounds. The first round comprehended filtering based on the metadata analysis like title, keywords, and abstract. As a result, we obtained 682 papers. We went more in-depth in the second round, analyzing the introduction, conclusion, and other sections where the authors explained their proposals. Finally, we got a total of 87 papers for extracting information to answer our RQs. To support all this process, we used Google Spreadsheets. The Table 1 presents the breakdown totals we got for each search engine of each filtering phase.

**Table 1.** Number of results per database/search engine.

| Search Tool | Results | $1^{st}$ Filter | $2^{nd}$ Filter |
|---|---|---|---|
| ACM DL | 337 | 57 | 03 |
| IEEE Xplore | 279 | 138 | 15 |
| Scopus | 361 | 159 | 22 |
| Springer Link | 1000 | 328 | 47 |
| Total | 1977 | 682 | 87 |

Two researchers participated in the searching, selection, data extraction, analysis, and synthesis stages. The first author made the searching, selection, data extraction, data analysis, and data synthesis tasks, and the second one revised the obtained results. The authors had weekly meetings where they could discuss the results of this study. There were no conflicts about the selection of studies.

## 5 Discussion and Results

This section presents an analysis of papers and the answers to each research question presented before. The subsection 5.1 presents an overview of the studies found, including references to all selected papers. The remaining subsections (5.2 to 5.9) are related to each one of the RQs, where papers were explored and analyzed in order to answer them.

### 5.1 Studies' Overview

First of all, we present, in Table 7 (Appendix A), a list with the 87 selected papers. The table shows the papers' references and publication venues.

### 5.2 RQ1: Distribution of Works per Year

Figure 2 presents a temporal distribution chart of the studies based on the year of publication extracted from each paper. We can observe that most works were published between 2006 and 2015.

There are a lower number of works from 2015. The work of Roda et al. (2018) showed that there is a significant amount of work on context-aware architectures since 2015.

[4]The search on SpringerLink returned 10387 results. Because of the enormous number of papers, just the 1000 most relevant ones have been selected.

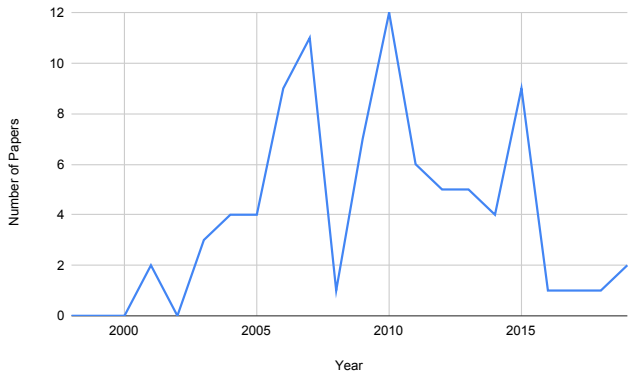


**Figure 2.** Articles per year.

However, these architectures were not applied in a smart city domain, are not distributed, or do not obtain data from a heterogeneous and distributed sensor arrangement (according to E6 exclusion criteria).

There is also a trend in the development of smart city solutions that use edge (Khan et al., 2020) and fog (Javadzadeh and Rahmani, 2020) computing paradigms. These two paradigms are inherently distributed, and context-awareness is one of the key requirements in these smart city infrastructures (Khan et al., 2020). There is a significant increase in research interest in smart city solutions based on fog and edge computing after 2015. Therefore, future works can adapt contemporary context recognition architectures or develop new ones that use edge and fog computing infrastructures.

## 5.3 RQ2: Most Evident Publication Venues

We extracted the name of the journal or conference where each paper has been published. The Table 2 shows the top publication venues. As we present in this table, around 21.8% of the 87 selected papers are published in these publication venues: *Personal and Ubiquitous Computing*, *IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, *IEEE International Conference on Pervasive Computing and Communications*, and *IFIP International Conference on Distributed Applications and Interoperable Systems*.

Fifty-six papers have been published in other journals and conference proceedings. Each of these venues contains only one of the analyzed papers, and because of this reason, we do not show them in the table. However, the reader can check these venues consulting the Table 7.

## 5.4 RQ3: Most Used Architectural Patterns

In order to provide an overview of software architecture patterns used for building context recognition subsystems, we analyzed the sections of the papers that described the architecture of each approach and extracted the architectural pattern used by each one. After going through all the 87 selected papers, we have identified 11 different patterns listed in Table 3 along with the selected papers that apply them. The patterns list used in this work is based on the list provided by Roda et al. (2018). Notice that one pattern has been selected by work. Some works applied more than one pattern, and the most prominent has been selected. *Multi-tier* approach is the most common used one, followed by *multi-agent*, *layered*, *service-oriented*, *broker* and *client-server* approaches.

**Table 2.** Number of papers published in the topmost publication venues.

| Publication | Type | Papers |
|---|---|---|
| Personal and Ubiquitous Computing | Journal | 6 |
| IEEE/IFIP Int. Conf. on Embedded and Ubiquitous Computing | Conference | 5 |
| IEEE Int. Conf. on Pervasive Computing and Communications | Conference | 4 |
| IFIP Int. Conf. on Distributed Applications and Interoperable Systems | Conference | 4 |
| European Conf. on Smart Sensing and Context | Conference | 2 |
| Int. Conf. on Computational Science and Its Applications | Conference | 2 |
| Int. Conf. on Context-Aware Systems and Applications | Conference | 2 |
| Int. Conf. on Grid and Pervasive Computing | Conference | 2 |
| Int. Conf. on Mobile Data Management | Conference | 2 |
| Multimedia Tools and Applications | Journal | 2 |

Multi-tier architectures focus on the decomposition of the design into logical components. These architectures usually have components to obtain data from sensors, components to aggregate/fuse these data, and components to perform context reasoning and distribution. Distributed and heterogeneous sensors can be attached to them in a plug-and-play manner. However, most multi-tier architectures present centralized components that aggregate context information from context sources and provide interfaces for applications. Some exceptions include the following works:

- Dey and Mankoff (2005) describe an architecture composed by three kinds of components: *widgets*, *interpreters* and *mediators*. There can be several instances of each component, and a distributed arrangement of the instances is possible. Widgets encapsulate access to sensors and store context information. Interpreters apply reasoning approaches to context data, and mediators are responsible for dealing with ambiguity issues. Widgets can receive data from physical sensors, other widgets, or interpreters.
- Malandrino et al. (2010) show a framework that uses a distributed profile management architecture. Three components/modules are responsible for addressing profile management: user profile management module, operator profile management module, and service provider management module. These modules send profile data to a centralized context provider module that aggregates the data and reasons on these data.
- Ranganathan and Campbell (2003) present an distributed architecture based on two types of components: *context providers* and *context synthesizers*. Distributed context providers are sources of low-level context data.

**Table 3.** Papers per Architectural Pattern.

| Arch. Pattern | Papers | Total |
|---|---|---|
| Multi-tier | S2, S3, S5, S13, S14, S18, S27, S31, S35, S38, S40, S50, S56, S60, S67, S69, S70, S74, S77, S78, S83, S84 | 22 |
| Multi-agent | S4, S10, S15, S21, S26, S28, S32, S37, S39, S41, S45, S47, S49, S54, S64, S68, S82, S86 | 18 |
| Layered | S1, S6, S11, S19, S20, S36, S48, S51, S52, S57, S71, S72, S76 | 13 |
| Service-oriented | S30, S33, S34, S44, S46, S61, S62, S63, S65, S80, S87 | 11 |
| Broker | S24, S42, S43, S53, S55, S59 | 6 |
| Client-server | S9, S12, S16, S17, S23, S79 | 6 |
| Other | S7, S8, S9, S12, S16, S17, S22, S25, S29, S58, S66, S73, S79, S81, S85 | 15 |

Context synthesizers generate high-level context information from context obtained by one or more context providers. Applications can obtain context data from providers, synthesizers, or both.

- Efstratiou et al. (2001) present an architecture where services provide context data. A context discovery component can detect available context services and access them. Agents encapsulate context inference mechanisms and can be plugged into the platform to incorporate new context recognition capabilities into the system.
- Ryu et al. (2007) show an architecture where context sources are distributed in several mobile and wearable devices. Context widgets are the data sources deployed in each device. A distributed process aggregates these data. Devices develop a local aggregation, and a coordinator aggregates data from each device. The coordinator is responsible to made context information available to applications.

A multi-agent architecture is composed of agents that can interact to complete a set of tasks. Agent-based architectures are inherently distributed. In general, the analyzed multi-agent architectures can be grouped in the following categories:

- *Teams* composed of a fixed number of agents with reasoning capabilities. Each agent performs a different role in the context recognition process. Specific distributed wrapper agents can get data from sensors or other context data sources. Most analyzed multi-agent architectures are organized in this manner (S4, S10, S15, S26, S28, S32, S37, S39, S45, S47, S49, S54).
- Multiple and distributed agents are organized in a *hierarchy*. Agents at the top receive requests from applications Agents in lower levels can obtain data from context sources and process them. This kind of architecture has been applied in the following works: S41, S82, S86.
- *Society* of agents where each one has a partial view of the context information. Each agent has inference mechanisms and can exchange context knowledge with other agents. Therefore, the context knowledge is distributed over all agents. There are rules and social laws that define how agents can communicate and exchange knowledge. Only three works presented such kind of agent organization (S21, S64, S68).

Layered architecture is an architecture where a different layer performs each context recognition step. These layers can be deployed in a centralized server or can be distributed on several devices. Common layers found in the majority of works are:

- *Sensing layer*, *Perception layer*, or *Acquisition layer*, that aims to get data from sensors and other raw context sources.
- *Aggregation layer* that is responsible for aggregating and fuse data from multiple data sources. After receiving the context data from diverse sources, the context aggregation layer would clean, pre-process, and apply sensor data fusion techniques to provide accurate and reliable context data to be processed further.
- *Inference or Reasoning layers* that are one or more layers responsible for applying inference techniques to recognize high-level context. This process can be divided into two or more layers and can be distributed and executed in parallel.

In service-based architectures, services encapsulate context acquisition and inference. We identified four types of services organizations among the analyzed works:

- Distributed Web services perform context acquisition from sensors and other data sources. Centralized Web services process these data and recognize the high-level context (see works S46, S61, S62, S65, and S87). The work of Yang et al. (2009) (reference S65) also includes an approach that selects a localization service based on the detected user context.
- Distributed Web services are organized in logic layers. Usually, web services for obtaining raw data from sensors are in the lowest layer, while services that can recognize high-level context are in the higher layers (see works S34, S63, and S80). There is no centralized service responsible for recognizing context. Bernardos et al. (2010) present a particular case where services are located in mobile nodes that can exchange context information (see reference S63).
- Network of context provider services. These services can contact each other to exchange context data. Services must register in a broker or service registry. This broker or registry makes it possible that services can find each other.

Broker architectures are characterized by a single access point that intermediate the communication between applications and context providers. Context providers are modeled

as plugins that can obtain raw data from sensors or have context inference capabilities. Plugins are different from agents and services because they do not have autonomy and are not distributed.

Client-server architectures use two types of component types: clients and servers. Clients interact by requesting services of servers, which provide a set of services. The analyzed client-server architectures use clients deployed at mobile devices to access context reasoning and storage services in servers. Servers in the cloud are the most common approach. These servers can use the high-capacity processing power of the cloud to process raw context data.

Finally, the remaining works use five other different architecture patterns. These patterns include peer-to-peer, publish/subscribe, and pipe-and-filter architectures.

The analysis of existing works shows a logical organization of architectural components according to context life cycle: acquisition components and reasoning components. The selected architectures logically organized these components in a set of layers or tiers. This kind of organization allows a clear separation of concerns between different components.

Acquisition components are often organized as plugins, so that context recognition systems can easily add new sensing capabilities. As stated before, distributed components, Web services, and agents are the most common implementations of acquisition components. Current smart city middlewares such as FIWARE(Cirillo et al., 2019), Sentilo[5], OpenIoT[6], and Arrowhead[7] provide service-based interfaces to provide data from IoT and other smart cities data sources. Therefore, a service-based architecture seems to be the most suitable to be used in the acquisition process.

Few works present a distributed organization of reasoning components. Most architectures provide centralized inference modules that aggregate data from distributed acquisition components and reason about these data to infer high-level context. Service-oriented and multi-agent systems are the most common paradigms in implementing distributed reasoning mechanisms provided by the selected works. These paradigms are naturally distributed and can provide a decentralized processing mechanism. Registries can provide a way to find services or agents that can provide some specific reasoning capabilities. These selection mechanisms can also provide mechanisms to select services or agents based on quality parameters (Huebscher and McCann, 2006).

Therefore, a combination of architectural patterns can provide the necessary separation of concerns and distribution of processing. Examples of such architectures are:

- layered architectures that logically organized a set of distributed services that provide acquisition capabilities from an IoT infrastructure and reasoning mechanisms to infer high-level contexts;
- multi-tier architectures where reasoning agents reason about low-level context data provided by Web services interfaces of smart city middlewares.

[5] `https://www.sentilo.io/wordpress/`
[6] http://www.openiot.eu/
[7] https://arrowhead.eu/1492

## 5.5 RQ4: Management of Context Information

As shown in section 2.2, context information have a four-step life cycle. We analyze sections of papers that describe in more detail the context information management process (acquisition, modeling, reasoning, and distribution). We extracted information about three aspects related to the context life cycle in the analyzed architectures:

- which context data sources are supported;
- how context has been modeled;
- what context distribution methods have been used.

Table 4 shows the context data source support per selected papers. The context sources shown in the table are those listed in Section 2.1. Note that the same job can use more than one type of data source.

It is possible to see that physical sensors in IoT infrastructures, smartphones, or wearable devices are the most commonly supported by context recognition architectures. Physical sensors are the most crucial real-time context data source in cities. Furthermore, as identified in Nascimento et al. (2021), physical sensors are also the most common source of raw data in context inference approaches.

We also analyzed how the data obtained from data sources have been modeled in the selected architectures. The results are shown in Table 5.

Even if several studies do not specify the model used, it is possible to notice that models based on ontology are the most popular. This fact agrees with many surveys in context-aware computing and sensor data management, which showed that ontologies are the preferred mechanism of managing and modeling context.

There are several reasons to develop and use ontologies in contrast to other modeling techniques. The most common reasons related to context recognition architectures are to share a common understanding of information structure among software components, analyze domain knowledge, separate domain knowledge from operational knowledge, enable reuse of domain knowledge, and high-level knowledge inferring (Perera et al., 2013). An ontology is an essential tool in a distributed structure to capture consensual knowledge accepted and understandable by a group. Therefore, the context knowledge can be easily shared and reused.

Finally, we analyzed as well how context information is distributed to context consumers. Table 6 summarizes the results. Each paper was classified in one of four categories: *query*, *publish/subscribe*, *both* (if both techniques were applied), or *not specified* (if the paper did not specify how the context information has been delivered).

Query is the most used technique. The most complex mechanisms use query languages. The languages used in the analyzed solutions vary from general-purpose ones (such as SPARQL and RDQL) to specific context query languages (see S24, S43, S45, and S52).

## 5.6 RQ5: Security & Privacy Issues

As seen in the previous section, context recognition architectures in smart cities commonly use IoT devices and mo-

**Table 4.** Papers per Data Source.

| Data Sources | Papers | Total |
|---|---|---|
| IoT | S1, S2, S3, S4, S5, S6, S7, S8, S10, S11, S13, S14, S15, S18, S19, S20, S22, S24, S25, S27, S28, S29, S30, S34, S35, S36, S37, S38, S39, S40, S41, S42, S43, S44, S45, S46, S47, S48, S49, S50, S51, S52, S53, S54, S55, S56, S57, S58, S59, S60, S61, S62, S63, S64, S65, S66, S67, S68, S69, S70, S71, S72, S73, S74, S75, S76, S77, S78, S79, S81, S82, S83, S84, S85, S86, S87 | 76 |
| Smartphone | S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S12, S13, S14, S16, S17, S18, S19, S20, S22, S23, S24, S26, S29, S30, S33, S34, S37, S38, S39, S40, S41, S43, S44, S45, S47, S48, S49, S50, S51, S52, S53, S55, S56, S58, S59, S63, S64, S66, S67, S68, S71, S72, S73, S75, S76, S77, S78, S79, S80, S81, S82 | 61 |
| Wearable | S1, S2, S3, S4, S5, S6, S7, S8, S13, S14, S17, S18, S19, S20, S22, S24, S25, S29, S34, S37, S38, S40, S41, S43, S44, S45, S47, S50, S51, S55, S58, S59, S63, S64, S66, S67, S68, S70, S72, S73, S75, S76, S77, S78, S79, S81, S82 | 47 |
| Web services | S2, S7, S8, S20, S29, S30, S31, S34, S40, S48, S55, S58, S66, S69, S70, S80, S81, S82 | 18 |
| Social Net. | S1, S2, S7, S8, S29, S34, S40, S48, S49, S53, S55, S58, S66, S69,S70, S82 | 16 |
| Local DBs | S6, S10, S17, S21, S29, S34, S37, S40, S44, S45, S55, S57, S58, S69, S76 | 15 |
| Open Data | S2, S7, S8, S29, S34, S40, S48, S55, S58, S66, S80 | 11 |
| Crowdsensing | S2, S7, S8, S29, S34, S40, S52, S55, S58, S66 | 10 |
| Other | S1, S6, S32, S34, S40, S55, S58 | 7 |

**Table 5.** Papers per context modeling approaches.

| Model | Papers | Total |
|---|---|---|
| Ontology | S1, S3, S6, S7, S8, S10, S11, S13, S15, S16, S21, S24, S25, S26, S28, S30, S31, S32, S34, S35, S39, S41, S45, S46, S47, S54, S58, S61, S62, S64, S65, S66, S68, S69, S77, S82, S87 | 37 |
| Markup | S22, S44, S51, S67, S81, S83 | 6 |
| Key-value | S2, S9, S55, S63, S70 | 5 |
| Object-based | S5, S12, S29, S33, S74 | 5 |
| Logic-based | S4, S27, S40, S56, S59 | 5 |
| Graphical | S50, S71, S79 | 3 |
| Not specified | S14, S17, S18, S19, S20, S23, S36, S37, S38, S42, S43, S48, S49, S52, S53, S57, S60, S72, S73, S75, S76, S78, S80, S84, S85, S86 | 26 |

**Table 6.** Papers per context delivery techniques.

| Del. Tech. | Papers | Total |
|---|---|---|
| Not spec. | S1, S4, S6, S7, S8, S9, S11, S13, S14, S15, S17, S18, S19, S20, S21, S22, S26, S27, S29, S31, S32, S38, S40, S48, S49, S56, S57, S60, S62, S65, S67, S69, S71, S73, S75, S77, S82 | 37 |
| Query | S2, S3, S5, S10, S12, S16, S33, S34, S36, S39, S46, S47, S54, S64, S68, S70, S72, S79, S83, S84, S87 | 21 |
| Both | S23, S24, S28, S30, S41, S42, S43, S44, S45, S50, S52, S55, S58, S61, S63, S66, S74, S76, S78, S80 | 20 |
| Pub./Subs. | S25, S35, S37, S51, S53, S59, S81, S85, S86 | 9 |

bile/wearable sensors as raw data sources. These devices generate a considerable amount of data. These data are often not centralized but distributed across different platforms or parties (data distributed in different devices owned by different companies/organizations). Furthermore, cloud and fog computing are popular paradigms used in smart cities' applications. These paradigms lead to context data transfer between distributed nodes and distributed processing of these data.

In this situation, context recognition architectures need to integrate these distributed data to provide high-level context information. However, these architectures often deal with personal user data, and the leaking of this information will be a considerable threat to the user's privacy and security.

These privacy and security issues complicate integrating distributed context data and a distributed processing of context data in a reliable way. Therefore, it becomes essential for a distributed context recognition architecture to protect users' and entities' private information when integrating the distributed context data.

We read again sections of the papers that describe the proposed architectures and sections where security and privacy strategies were described. We extracted information from each paper about the usage technologies and approaches related to security and privacy. Only ten of the 87 analyzed papers presented some security/privacy approach. The techniques applied in these approaches are summarized below:

- *Authentication*, where client applications authenticate users to have access to context provider modules (S23, S49, S51, S75). These modules can be context repositories deployed in cloud servers (S23, S45) or available in distributed modules with access to physical sensors or other raw data sources (S23, S51, S75). Thus, user authentication mechanisms are implemented in applications and are outside context-recognition modules or middlewares. An alternative that context-recognition

subsystems can implement is the use of certificates to control access to context databases in cloud servers or distributed context providers (S23).

- *Privacy and security policies* specified by rules (S8, S49, S63, S72, and S74). These rules define if a user, application, or another context consumer can have full access or partial access to some piece of information in a given context. Rules can be defined using concepts and relationships semantically defined in an ontology (S8) or specific languages used to privacy policies specification (S72, S74). Specific protocols such as P3P[8] are an option to guide the authorization checking process (S72).
- *Cryptography and other network security approaches*. Context data and personal user information can be obfuscated using one-way hashing or encrypted by a two-way cryptography method (S1, S23, and S49). S1 uses *Advanced Encryption Standard* (AES) for context information storage. Some works apply data randomization techniques and one-way hashing for personal information obfuscation during context data transfer (S1, S23).
- *Other techniques*, such as the execution of plugins in a sandbox (S75) or an explicit validation of context sources by applications (S52).

## 5.7 RQ6: Decentralized, Resilient, and Dynamic Composed Architectures with Fusion and Plugin Support Capabilities

The RQ6 led to an analysis of which papers presented an architecture with one or more of the five characteristics related to this work ( decentralization, resilience, dynamic composition, plugin support, and context data fusion). We analyze the sections describing the proposed architectures to extract the presence or not of each characteristic and which technologies each paper used in the implementation of these characteristics. Fifteen papers did not present any characteristics (S1, S4, S5, S6, S11, S14, S15, S16, S17, S26, S35, S49, S70, S71, S81).

The remaining 72 papers are presented in Figure 3. The papers are arranged in a Venn diagram where each characteristic is shown as a set. Each work reference is presented inside the areas corresponding to the intersection of related characteristics.

A full reading of each of these 72 articles was carried out to identify the five mentioned characteristics in each architecture. The results obtained are as follows:

- *Plugin Support*: it is a general characteristic of the analyzed architectures, presented in about 74 % of the 87 analyzed. These plugins include new data sources (usually physical sensors) and new reasoning capabilities attached to the architecture to improve its functionalities. Web services, multi-agent systems, and plug-and-play components are the leading technologies used to implement plugins.
- *Decentralization*: about 36 % of the analyzed architectures presented a decentralized process of context recognition. In general, these architectures use one of three arrangements of components: layered, peer-to-peer, or registry-based. A layered organization contains upper layer components or nodes responsible for receiving applications' requests and contact components in lower layers to obtain context information. Usually, these architectures use a two-layer organization, where the lowest layer obtains data from sensors and the highest layer process these data to obtain high-level context information. In peer-to-peer organizations, distributed nodes obtain data from local sensors and process them to obtain context information. Nodes can share their data in order to improve their knowledge about the context. Registry-based architectures use a set of registries or directory facilitators where context providers can subscribe. Applications access these registries in order to find context providers.
- *Dynamic Composition*: around 25% of the works present a solution that uses a dynamic and automatic composition of inference mechanisms. The most common approach for dynamic composition is to select context providers and reasoning mechanisms from a query or context requirements specified by an application (S13, S24, S34, S42, S43, S44, S47, S50, S52, S55, S61, S80, S87). The capabilities informed by each provider/reasoner are crucial in this selection process. It is possible to improve the selection process using QoC parameters (S13, S42). Another approach consists of self-organization mechanisms, where nodes or agents can announce their capabilities, discover other nodes, and use a contract mechanism to organize contract chains or exchange knowledge to create advanced and complex compound context recognition services (S7, S8, S21, S25, S33, S64, S66). Finally, the other two works use different composition approaches. S22 uses an approach based on rules to select specific recognition plans when some event occurs. In S59, applications and other context consumers execute a multicast process to identify and select context providers.
- *Data Fusion*: about 12% of the studied architectures presented some fusion mechanism during the context recognition process. The works presented in S9 and S72 use a fusion approach based on a reliability/confidence index to resolve conflicts in data obtained from different context providers. S18 fuses sensor data from distributed homogeneous sources using Dempster-Shafer's theory to lead with inconsistent raw data. Rules are applied to resolve conflicts and fuse context information in S31. The remaining works do not present sufficient details about their fusion processes.
- *Resilience*: only 8% of the studies had any fault tolerance mechanism. The most used approach presents an architecture where context providers can substitute other equivalent ones in case of a failure (S28, S42, S52, S82). Another approach uses context information sharing among distributed nodes or agents that make the architecture adaptable in case of a node failure (S24, S32). The work presented in S62 used a self-configuring algorithm to generate adaptation in case of a sensor failure.

Of all the works, only 4 (4.6 %) have a combination of

[8] https://www.w3.org/P3P/

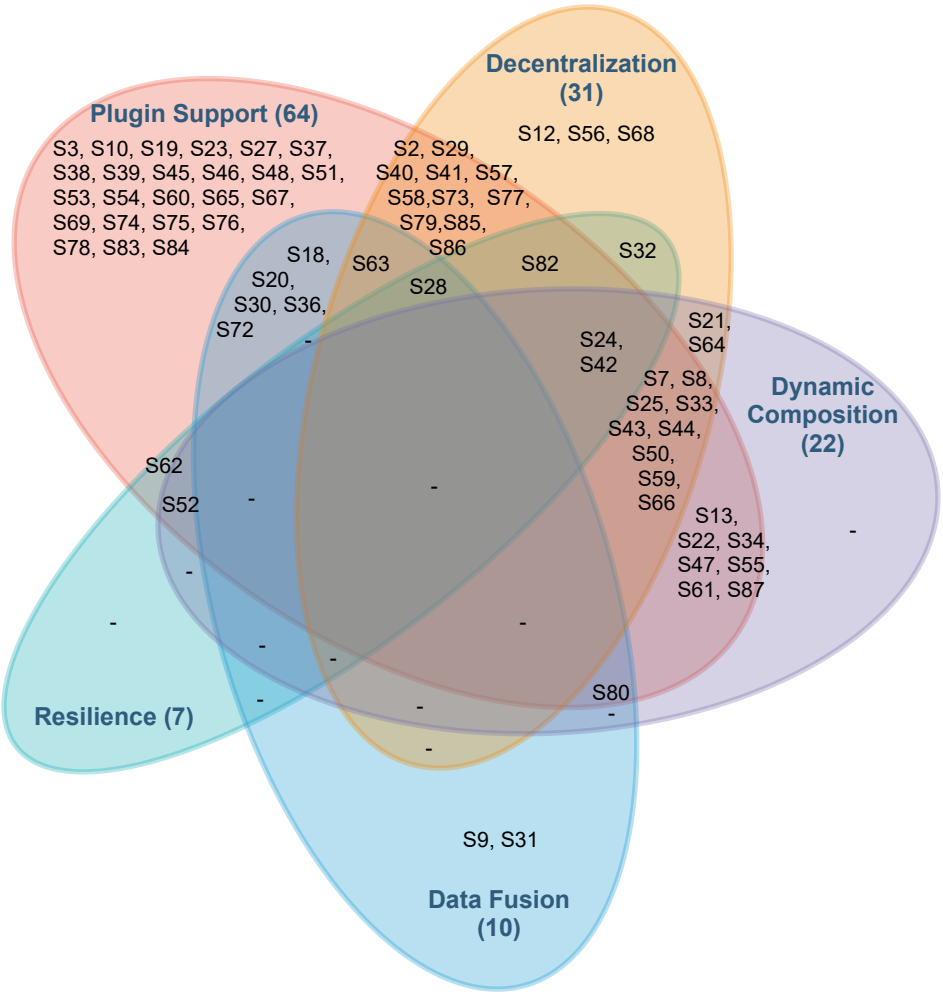


**Figure 3.** Venn diagram of articles per research area. The numbers after each areas' names indicate the number of papers identified in each of them.

decentralized architecture, the possibility of adding sensors, and resilience. We present a brief description of each work below.

The work S24 (Geihs and Wagner, 2012) presents a middleware architecture proposal for context management designed to be used within a larger project called MUSIC (*(Self-Adapting Applications for Mobile Users in Ubiquitous Computing Environments*). The platform allows the addition of new sensors and inference mechanisms as *plugins*. New sensors can be added in a *on-thef-fly* way, as they are discovered. An automated mechanism enables or disables *plugins* according to the needs of the application. These *plugins* can be available in repositories and can be easily reused (so the architecture supports dynamic composition). The architecture allows these plugins to be organized into distributed nodes. These nodes can exchange context information with each other. Plugins are treated as services and selected by a negotiation process based on QoS (*Quality of Service* - Quality of Service). The architecture does not have an apparent fault tolerance mechanism (the distribution of context information on distributed nodes can guarantee some degree of resilience). It also does not offer any precise mechanism for data fusion or validation of context information. Also, the work does not address any privacy or security issues.

S28 (Sebbak et al., 2010) features a multi-agent architecture integrated with a lower layer, composed of services responsible for obtaining data from sensors. Four categories of agents compound the society: the context service agents, the context aggregation agents, the context knowledge agents, and the context inference agents. These agents use the directory agent to register services and search for a specific context service provider. The context service agent can acquire contexts from OSGi services or directly from ubiquitous sensors located in the environment. Context services can provide other agents with contextual knowledge, such as the user's location or the weather in a specific location. The aggregation agent is dedicated to the fusion of contextual knowledge collected from various context service agents. When a context provider is unavailable, the aggregation agent can replace the unavailable service with another similar service without suspending its behavior. The context knowledge agent stores the context ontology and history of the contexts for later use. A set of services organized in an OSGi structure obtain sensor data. The architecture presented is decentralized, resilient,

extensible by adding new sensors and allows data fusion to form context information. However, the architecture does not support dynamic composition.

S42 (Huebscher and McCann, 2006) features a middleware organized in four layers. The bottom layer consists of physical sensors in an IoT infrastructure. The next layer consists of context providers (CPs) that aggregate and interpret the raw data from one or more sensors. When a CP enters the network, it announces itself to a directory service (DS). In the first contact, the CP informs the DS of the type of context it can provide and attaches QoC attributes' values that describe this provision. However, CPs are hidden from an application, introducing an extra layer of abstraction, the context service (CS). Context services retrieve context information from CPs on behalf of applications. This abstraction layer is useful because if a different CP is better for an application than the one currently used, the CS can adapt itself autonomously, changing to the best alternative without involving the application. The quality of a context provider is assessed through a utility function, informed by the applications. Another useful feature is that if a CP fails, it can be replaced by another related to the same CS. A context service is responsible for handling a specific context category. In short, the proposed middleware allows for some decentralization (several directory services can run in parallel), resilience, and dynamic composition. However, the architecture does not present a clear data fusion mechanism.

S82 (Alvarez-Napagao et al., 2014) proposes an architecture that uses smart mobile devices (like *smartphones*) to make possible the concept of *human as a sensor*. Thus, the proposed work aims to: (i) improve the knowledge obtained through data sources; (ii) detecting unexpected situations in cities that can affect large groups of people; (iii) making services available to users who can exploit the knowledge generated (such as recommendation systems, for example). For this, the work proposes an architecture called *SUPERHUB*. This architecture has a semantic interpreter responsible for providing context knowledge through RDF tuples. Internally, several agents compound the semantic interpreter. Tracking agents assign themselves an API or *web service* as a source of context data and manage the receipt of data from them. Work agents are responsible for aggregating tracker data and producing RDF triples with high-level context knowledge. These aggregation processes can be added or removed at run time. All agents can be distributed. Another agent can replace an agent that fails through an agent *pool* managed by the semantic interpreter. Therefore, the architecture described has a certain decentralization level (although a centralized semantic interpreter stores the context information) and resilience. Sensors can be added via new tracking agents. Data fusion mechanisms can be included in worker agents, although the architecture does not provide anything specific. Besides, no dynamic composition mechanism is envisaged.

## 5.8 RQ7: Validation Methods

To answer RQ7, we read the evaluation section of each paper to extract the validation types realized by each approach in the set of papers. The classes of validation were based on the classes presented in Machado et al. (2018). The classes of validation are:

- *Observational Study* is an unobtrusive method of gather observations to support a hypothesis, often taken by a survey statistically. Five of the analyzed papers used an observational study for validation: S3, S24, S43, S56, S70.
- *Simulation* is the method that uses software to generate data to mock a behavior in the real world. The works S11, S19, and S28 use simulations to validate their proposals.
- *Case Study* is the treatment of one or more determined cases and the collection of data generated by the approach. It is the most used approach identified in 26 papers: S2, S4, S5, S8, S10, S12, S14, S21, S22, S25, S26, S27, S30, S32, S34, S35, S36, S45, S50, S53, S54, S55, S59, S62, S68, and S76.
- *Experience Report* is the treatment applied to one specific case, but no particular effort is applied in controlling the context. Papers S9 and S17 use this validation approach.
- *Experiment* is an execution applied under control to observe the effects, usually using a dataset. The behavior of the system is measured and statistically validated. Four papers use an experiment for validation: S7, S73, S86, and S87.

Twenty-one papers used other validation methods (S1, S18, S23, S29, S31, S33, S39, S40, S41, S44. S52, S61, S63, S66, S77, S78, S81, S82, S83, S84, S85). The most used one is the implementation of applications to demonstrate the potential of a framework or middleware. Twenty-six papers do not present any validation descriptions (S6, S13, S15, S16, S20, S37, S38, S42, S46, S47, S48, S49, S51, S57, S58, S60, S64, S65, S67, S69, S71, S72, S74, S75, S79, and S80).

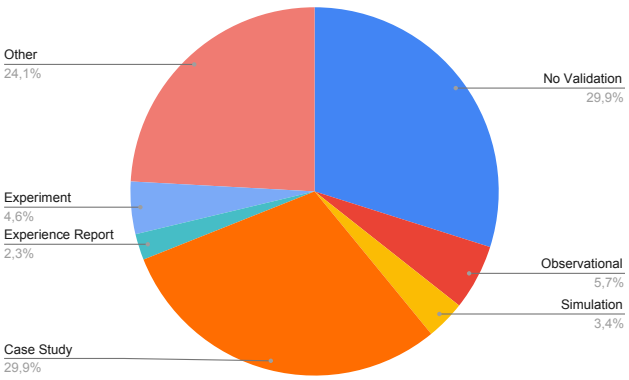


**Figure 4.** Validation methods used in the analyzed works.

Figure 4 summarizes the use of validation methods among the analyzed papers. Case study is the most popular method. However, there are a high number of possible scenarios of application for a context recognition architecture. Besides that, several works do not present any validation method or use other methods, especially the implementation of example applications to demonstrate the potential of a framework or middleware in the development of new context-aware solutions.

### 5.9 RQ8: Research Opportunities

RQ8 is based on the analysis performed previously in other RQs. In this subsection, it is sought to summarize a series of common issues found in the analyzed works and to present future directions of research:

1. *Decentralized, extensible and fault-tolerant architectures*. The literature review developed in this work identified several distributed architectures for context recognition suitable for smart environments like smart cities. Smart cities contain a huge number of distributed context data sources. Context recognition architectures in smart cities should be extensible. They should allow the easy addition of new sensors and context inference capabilities. They should also take advantage of new paradigms such as fog and edge computing to decentralize context processing. Furthermore, these architectures can use fault-tolerance mechanisms to deal with failures in nodes and make themselves resilient. However, only four of the analyzed architectures are decentralized, extensible, and fault-tolerant. Therefore, there is a research gap in developing context recognition architectures in smart cities that incorporate these features in a unique approach.
2. *Resilience*. Only seven of the 87 analyzed papers presented a resilient context recognition architecture. Most of the works focused on dealing with distributed context sources and how to manage context information. Recent smart cities' solutions use redundant physical sensors deployed in an IoT infrastructure. New context recognition architectures in smart cities can take advantage of this redundancy to provide resilient solutions. Furthermore, distributed solutions potentiate the development of redundant nodes or agents that can substitute each other in case of failure. Therefore, there is a research opportunity to explore resilience in these architectures.
3. *Context fusion*. Sensors data fusion is an interesting approach in context recognition when multiple sensors provide data about the same entity property. Techniques like Dempster-Shafer Theory or filters can fusion these data to improve the resulting context information. Advanced information fusion techniques can be applied to fuse context information provided by distributed redundant or complementary context providers. However, only 12% of the analyzed papers use data or information fusion in context processing. It will take more effort to explore data and information fusion techniques in context recognition applied to smart cities to reach a good maturity level.
4. *Security and privacy*. Context-aware systems often deal with users' personal information. Distributed context recognition should ensure some security and privacy level so that the user's data is not accessed for malicious purposes. Only ten of the analyzed works presented some approaches to their users' data security or privacy. Therefore, there is a research gap exploring security and privacy approaches for distributed context recognition architectures in smart cities. The analyzed works have applied techniques such as authentication of users and applications, privacy policies specified in rules, data obfuscation, and cryptography. There is a research opportunity to investigate further how these and other methods can be applied in context recognition architectures to ensure security and privacy. There are challenges involved in developing such solutions. Resource-restricted devices such as smartphones and wearables are extensively used in context recognition solutions in smart cities. Privacy-preserving context-aware technologies are still an open subject for these devices given that such solutions often impose to utilize computational expensive and energy harvesting algorithms (Yürür et al., 2014).

## 6 Threats to Validity

This mapping study addresses the following threats to validityAmpatzoglou et al. (2019):

- *Adequacy of initial relevant publication identification* that is related basically to problems that might occur when the researchers are building the search string and problems that can arise from using very specific, too broad, or not credible search engines. To mitigate this threat, we started using a strategy to construct the search string where we selected keywords from the research objectives and related synonyms. We also performed pilot searches to train our search string and selected known digital libraries (ACM, IEEE, Springer, and Scopus) to perform the search for studies. The search outcomes were documented in a spreadsheet using Google Sheets.
- *Limited number of journals and conferences*: to avoid this threat, we use a broad search process in generic search engines.
- *Paper inaccessibility*: the number of studies with missing full texts was low compared to the population (1.6% of papers selected after the first filtering).
- *Handling of duplicate articles*: to mitigate this threat, we excluded older duplicates with the same title, same authors, same year, and same publication venue. We also exclude older publications from the same authors about the same approach.
- *Inclusion/exclusion of grey literature*: we exclude grey literature and selected only papers published in conference proceedings and journals.
- *Study inclusion/exclusion*: the inclusion and exclusion criteria were well defined and documented in the search protocol. We used pilot iterations to revise these criteria. All authors revised all criteria.
- *Data extraction bias*: to mitigate this threat, we perform a random paper screening to cross-check data extraction.
- *Bias of classification schema*: we continuously updated the classification schema, until it becomes stable. All primary studies were classified in one or more classes considering different aspects (such as architectural patterns and validation methods).
- *Generalizability*: to mitigate this threat, we applied a broad search process without an initial start date. We check other existing studies, and our findings comply with the results presented by them. For example, our

analysis of architectural patterns complies with the analysis presented by Roda et al. (2018) related to context-aware systems in general. Security and privacy lacking support is also a result that complies with recent studies that put them as open issues (Talari et al., 2017).

# 7 Conclusion

This paper presents a systematic mapping of distributed context recognition architectures suitable for use in the smart city domain. We considered architectures that contain distributed processing or that use physically distributed context sources. The research's main goal was to analyze such approaches, provide an overview of state of art, and suggest directions to the research community. The search for papers used four search engines, and after two filterings, 87 papers were selected for analysis.

The results and analysis presented in Sect. 5 have contributed to identifying the research area and lead us to draw some conclusions that are summarized below:

- The most used architectural patterns were multi-tier, multi-agent, layered, service-oriented, broker, and client-server. Components, agents, layers, and services can be deployed in distributed physical nodes to execute different reasoning mechanisms or obtain data from nearby sensors. These architectural patterns allow those modules to have cohesion and functional independence, contributing to the reuse of components and easier maintenance.
- Physical sensors are the primary source of raw context data. These sensors are usually deployed in an IoT infrastructure, a smartphone, or a wearable device.
- Ontologies are the most used modeling approaches. In distributed architectures, they allow a shared understanding of domain concepts and a high-level knowledge inferring.
- The analyzed architectures are, in general, extensible. New sensors and other raw context data sources can be easily aggregated. New context reasoning capabilities can be added to the architectures on the fly as plugins.
- Decentralization, dynamic composition, resilience, and data fusion are still less frequent concepts implemented in context recognition architectures. They represent essential features that can significantly improve the robustness, scalability, and efficiency of these architectures.

Some open issues were identified from the analysis of the papers and were detailed in 5.9. These issues are summarized below.

- There is a research gap in developing architectures that integrate decentralization, plugin support, and resilience. Resilience, in particular, is a feature present in a few analyzed works.
- Data and information fusion techniques can improve context recognition architectures in smart cities where an increasing number of redundant and complementary sensors are available. These techniques are present in a few analyzed papers.
- Security and privacy mechanisms are still lacking in context recognition architectures. There is another research opportunity in the implementation of these mechanisms in future smart cities context recognition architectures.

Smart cities are a hot topic nowadays. The analysis presented in this paper will support future context recognition architectures development in this domain. As future works, we expect to investigate data and information fusion techniques suitable for distributed context recognition in smart cities and develop an architecture that incorporates decentralization, plugin support, dynamic composition, resilience, and data fusion.

# A Selected Papers

The 87 papers included in this mapping study are listed in Table 7.

# References

Abkenar, A. B., Loke, S. W., Zaslavsky, A., and Rahayu, W. (2019). Garsaaas: group activity recognition and situation analysis as a service. *Journal of Internet Services and Applications*, 10(1):5.

Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P. (1999). Towards a better understanding of context and context-awareness. In Gellersen, H.-W., editor, *Handheld and Ubiquitous Computing*, pages 304–307, Berlin, Heidelberg. Springer Berlin Heidelberg.

Achilleos, A. P. and Kapitsaki, G. M. (2014). Enabling cross-platform mobile application development: a context-aware middleware. In *International Conference on Web Information Systems Engineering*, pages 304–318. Springer.

Aguilar, J., Jerez, M., Mendonca, M., and Sánchez, M. (2016). Misci: autonomic reflective middleware for smart cities. In *International Conference on Technologies and Innovation*, pages 241–253. Springer.

Al Nuaimi, E., Al Neyadi, H., Mohamed, N., and Al-Jaroodi, J. (2015). Applications of big data to smart cities. *Journal of Internet Services and Applications*, 6(1):1–15.

Al-Sultan, S. and Zedan, H. (2017). A unifying architecture model for developing context-aware systems. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 1–7. IEEE.

Albino, V., Berardi, U., and Dangelico, R. M. (2015). Smart cities: Definitions, dimensions, performance, and initiatives. *Journal of urban technology*, 22(1):3–21.

Alegre, U., Augusto, J. C., and Clark, T. (2016). Engineering context-aware systems and applications: A survey. *Journal of Systems and Software*, 117:55–83.

Alvarez-Napagao, S., Tejeda-Gómez, A., Oliva-Felipe, L., Garcia-Gasulla, D., Codina, V., Gómez-Sebastia, I., and

**Table 7.** Final list of selected papers.

| ID | Reference | Publication Venue |
|---|---|---|
| S1 | Banos et al. (2015) | International Conference on Pervasive Computing Technologies for Healthcare |
| S2 | Dey and Mankoff (2005) | ACM Transactions on Computer-Human Interaction |
| S3 | Guo et al. (2011) | Personal and Ubiquitous Computing |
| S4 | Chun-dong et al. (2009) | International Conference on Natural Computation |
| S5 | Park and Lee (2005) | ACIS International Conference on Computer and Information Science |
| S6 | Malek et al. (2006) | IEEE International Conference on Pervasive Computing and Communications |
| S7 | Bicocchi et al. (2010) | IEEE Transactions on Systems, Man, and Cybernetics |
| S8 | Pu et al. (2010) | International Conference on Computational Intelligence and Security |
| S9 | Hegde et al. (2009) | IEEE Transactions on Multimedia |
| S10 | Qin et al. (2007) | Tsinghua Science and Technology |
| S11 | Anand (2015) | IEEE International Symposium on Nanoelectronic and Information Systems |
| S12 | Martin et al. (2011) | FTRA International Conference on Multimedia and Ubiquitous Engineering |
| S13 | Beamon and Kumar (2010) | IEEE International Conference on Pervasive Computing and Communications |
| S14 | Madhusudanan et al. (2010) | International conference on Computing, Communication and Networking Technologies |
| S15 | Furno et al. (2011) | IEEE Symposium on Intelligent Agent |
| S16 | Devaraju and Hoh (2008) | International Symposium on Information Technology |
| S17 | Rahman et al. (2010) | IEEE Instrumentation & Measurement Technology Conference Proceedings |
| S18 | Dargie (2007) | IEEE International Conference on Pervasive Computing and Communications |
| S19 | Abkenar et al. (2019) | Journal of Internet Services and Applications |
| S20 | Al-Sultan and Zedan (2017) | IEEE SmartWorld Ubiquitous Intelligence and Computing, Advanced and Trusted Computed, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People and Smart City Innovation |
| S21 | Haque et al. (2016) | International Conference on Context-Aware Systems and Applications |
| S22 | Sabagh and Al-Yasiri (2015) | Computer Science - Research and Development |
| S23 | Ferreira et al. (2015) | Frontiers in ICT |
| S24 | Geihs and Wagner (2012) | International Conference on Context-Aware Systems and Applications |
| S25 | Muro et al. (2012) | Medical and Biological Engineering and Computing |
| S26 | Yılmaz and Erdur (2012) | Expert Systems with Applications |
| S27 | Yu et al. (2010) | IEEE/IFIP International Conference on Embedded and Ubiquitous Computing |
| S28 | Sebbak et al. (2010) | International Conference on Machine and Web Intelligence |
| S29 | Martín et al. (2010) | International Conference on Pervasive Computing and Applications |
| S30 | Iglesias et al. (2010) | IEEE/IFIP International Conference on Embedded and Ubiquitous Computing |
| S31 | Malandrino et al. (2010) | Personal and Ubiquitous Computing |
| S32 | Zhou et al. (2009) | International Conference on Parallel and Distributed Systems |
| S33 | Presecan and Tomai (2009) | IEEE International Conference on Intelligent Computer Communication and Processing |
| S34 | Schmidt et al. (2009) | IEEE International Conference on Pervasive Computing and Communications |
| S35 | Lee (2007) | Symposium on Human Interface and the Management of Information |
| S36 | Tadj and Ngantchaha (2006) | International Conference on Mobile Technology, Applications and Systems |
| S37 | Zheng et al. (2006) | International Conference on Grid and Cooperative Computing |
| S38 | Yau et al. (2004) | International Computer Software and Applications Conference |
| S39 | Sashima et al. (2003) | International Workshop on Multi-Agents for Mass User Support |
| S40 | Ranganathan and Campbell (2003) | Personal and Ubiquitous Computing |
| S41 | Thomson and Terzis (2012) | IFIP International Conference on Distributed Applications and Interoperable Systems |
| S42 | Huebscher and McCann (2006) | Personal and Ubiquitous Computing |
| S43 | Paspallis and Papadopoulos (2014) | Personal and Ubiquitous Computing |
| S44 | Knappmeyer et al. (2009) | European Conference on Smart Sensing and Context |
| S45 | Qin et al. (2006) | International Conference on Grid and Pervasive Computing |
| S46 | Xu et al. (2013) | International Conference on Distributed, Ambient, and Pervasive Interactions |

*continued on next page*

*continued from previous page*

| ID | Reference | Publication Venue |
|---|---|---|
| S47 | Ngo et al. (2004) | IEEE/IFIP International Conference on Embedded and Ubiquitous Computing |
| S48 | Oliveira et al. (2015) | International Workshop on Multiagent Foundations of Social Computing |
| S49 | Maciel et al. (2014) | International Workshop on Collaborative Agents, Research and Development |
| S50 | Poulcheria and Costas (2012) | Central European Journal of Computer Science |
| S51 | Indulska et al. (2003b) | International Conference on Mobile Data Management |
| S52 | Riva (2006) | International Conference on Distributed Systems Platforms and Open Distributed Processing |
| S53 | Achilleos and Kapitsaki (2014) | International Conference on Web Information Systems Engineering |
| S54 | Konstantinou et al. (2010) | Multimedia Tools and Applications |
| S55 | Fonteles et al. (2013) | International Symposium on Web and Wireless Geographical Information Systems |
| S56 | Jiang et al. (2004) | International Conference on Pervasive Computing |
| S57 | Lu et al. (2007) | European Conference on Smart Sensing and Context |
| S58 | Saleemi et al. (2011) | International Conference on Next Generation Wired/Wireless Networking |
| S59 | Van Kleek et al. (2006) | International Symposium on Ubiquitious Computing Systems |
| S60 | Lee et al. (2006) | International Conference on Computational Science |
| S61 | Kim and Choi (2007a) | International Conference on Ubiquitous Intelligence and Computing |
| S62 | Cioara et al. (2009) | International Conference on E-Business and Telecommunications |
| S63 | Bernardos et al. (2010) | International Conference on Hybrid Artificial Intelligence Systems |
| S64 | Aguilar et al. (2016) | International Conference on Technologies and Innovation |
| S65 | Yang et al. (2009) | Machine Learning in Cyber Trust |
| S66 | Gu et al. (2007) | IEEE/IFIP International Conference on Embedded and Ubiquitous Computing |
| S67 | Efstratiou et al. (2001) | International Conference on Mobile Data Management |
| S68 | Sorici et al. (2015) | International Conference on Practical Applications of Agents and Multi-Agent Systems |
| S69 | Colace et al. (2015) | Data Management in Pervasive Systems |
| S70 | Rahman et al. (2014) | Multimedia Tools and Applications |
| S71 | Klimek and Rogus (2014) | International Conference on Artificial Intelligence and Soft Computing |
| S72 | Indulska et al. (2003a) | IFIP International Conference on Distributed Applications and Interoperable Systems |
| S73 | Riboni (2019) | CCF Transactions on Pervasive Computing and Interaction |
| S74 | Shim (2006) | International Conference on Computational Science and Its Applications |
| S75 | Cassens et al. (2013) | International Joint Conference on Ambient Intelligence |
| S76 | Conan et al. (2007) | IFIP International Conference on Distributed Applications and Interoperable Systems |
| S77 | Ryu et al. (2007) | IFIP International Workshop on Software Technologies for Embedded and Ubiquitous Systems |
| S78 | Guo et al. (2013) | Personal and Ubiquitous Computing |
| S79 | Roussaki et al. (2007) | Journal of Network and Systems Management |
| S80 | Mitschang et al. (2005) | Data Management in a Connected World |
| S81 | Pantsar-Syväniemi (2011) | International Conference on Grid and Pervasive Computing |
| S82 | Alvarez-Napagao et al. (2014) | Agent Technology for Intelligent Mobile Services and Smart Societies |
| S83 | Park et al. (2005) | IEEE/IFIP International Conference on Embedded and Ubiquitous Computing |
| S84 | Broens et al. (2007) | Meeting of the European Network of Universities and Companies in Information and Communication Engineering |
| S85 | de López Ipiña and Lo (2001) | IFIP International Conference on Distributed Applications and Interoperable Systems |
| S86 | Boytsov and Zaslavsky (2011) | Smart Spaces and Next Generation Wired/Wireless Networking |
| S87 | Kim and Choi (2007b) | International Conference on Computational Science and Its Applications |

Vázquez-Salceda, J. (2014). Urban context detection and context-aware recommendation via networks of humans as sensors. In *Agent Technology for Intelligent Mobile Services and Smart Societies*, pages 68–85. Springer.

Ampatzoglou, A., Bibi, S., Avgeriou, P., Verbeek, M., and Chatzigeorgiou, A. (2019). Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. *Information and Software Technology*, 106:201–230.

Anand, P. (2015). Enabling context-aware computing in internet of things using m2m. In *2015 IEEE International Symposium on Nanoelectronic and Information Systems*, pages 219–224. IEEE.

Attard, J., Scerri, S., Rivera, I., and Handschuh, S. (2013). Ontology-based situation recognition for context-aware systems. In *Proceedings of the 9th International Conference on Semantic Systems*, I-SEMANTICS '13, pages 113–120. ACM.

Banos, O., Amin, M. B., Khan, W. A., Ali, T., Afzal, M., Kang, B. H., and Lee, S. (2015). Mining minds: an innovative framework for personalized health and wellness support. In *2015 9th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*, pages 1–8. IEEE.

Bass, L., Clements, P., and Kazman, R. (2012). *Software architecture in practice: Third Edition*. Addison-Wesley Professional.

Batista, C., Silva, P. V., Cavalcante, E., Batista, T., Barros, T., Takahashi, C., Cardoso, T., Neto, J. A., and Ribeiro, R. (2018). A middleware environment for developing internet of things applications. In *Proceedings of the 5th Workshop on Middleware and Applications for the Internet of Things*, pages 41–46.

Bazire, M. and Brézillon, P. (2005). Understanding context before using it. In *International and Interdisciplinary Conference on Modeling and Using Context*, pages 29–40. Springer.

Beamon, B. and Kumar, M. (2010). Hycore: Towards a generalized hierarchical hybrid context reasoning engine. In *2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pages 30–36. IEEE.

Bernardos, A. M., Madrazo, E., and Casar, J. R. (2010). An embeddable fusion framework to manage context information in mobile devices. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 468–477. Springer.

Bicocchi, N., Baumgarten, M., Brgulja, N., Kusber, R., Mamei, M., Mulvenna, M., and Zambonelli, F. (2010). Self-organized data ecologies for pervasive situation-aware services: The knowledge networks approach. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(4):789–802.

Boytsov, A. and Zaslavsky, A. (2011). Ecstra–distributed context reasoning framework for pervasive computing systems. In *Smart Spaces and Next Generation Wired/Wireless Networking*, pages 1–13. Springer.

Broens, T., Quartel, D., and Van Sinderen, M. (2007). Towards a context binding transparency. In *Meeting of the European Network of Universities and Companies in Information and Communication Engineering*, pages 9–16. Springer.

Cassens, J., Schmitt, F., and Herczeg, M. (2013). Cake–distributed environments for context-aware systems. In *International Joint Conference on Ambient Intelligence*, pages 275–280. Springer.

Chun-dong, W., Xiao-qin, L., and Huai-bin, W. (2009). A framework of intelligent agent based middleware for context aware computing. In *2009 Fifth International Conference on Natural Computation*, volume 6, pages 107–110. IEEE.

Cioara, T., Anghel, I., and Salomie, I. (2009). A self-configuring middleware solution for context management. In *International Conference on E-Business and Telecommunications*, pages 332–345. Springer.

Cirillo, F., Solmaz, G., Berz, E. L., Bauer, M., Cheng, B., and Kovacs, E. (2019). A standard-based open source iot platform: Fiware. *IEEE Internet of Things Magazine*, 2(3):12–18.

Colace, F., De Santo, M., Moscato, V., Picariello, A., Schreiber, F. A., and Tanca, L. (2015). Patch: A portable context-aware atlas for browsing cultural heritage. In *Data Management in Pervasive Systems*, pages 345–361. Springer.

Conan, D., Rouvoy, R., and Seinturier, L. (2007). Scalable processing of context information with cosmos. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 210–224. Springer.

Dargie, W. (2007). The role of probabilistic schemes in multisensor context-awareness. In *Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW'07)*, pages 27–32. IEEE.

de López Ipiña, D. and Lo, S.-L. (2001). Sentient computing for everyone. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 41–54. Springer.

Devaraju, A. and Hoh, S. (2008). Ontology-based context modeling for user-centered context-aware services platform. In *2008 International Symposium on Information Technology*, volume 2, pages 1–7. IEEE.

Dey, A. K. and Mankoff, J. (2005). Designing mediation for context-aware applications. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 12(1):53–80.

Efstratiou, C., Cheverst, K., Davies, N., and Friday, A. (2001). An architecture for the effective support of adaptive context-aware applications. In *International Conference on Mobile Data Management*, pages 15–26. Springer.

Fernandez-Anez, V. (2016). Stakeholders approach to smart cities: A survey on smart city definitions. In *International conference on smart cities*, pages 157–167. Springer.

Ferreira, D., Kostakos, V., and Dey, A. K. (2015). Aware: mobile context instrumentation framework. *Frontiers in ICT*, 2:6.

Fonteles, A. S., Neto, B. J., Maia, M., Viana, W., and Andrade, R. M. (2013). An adaptive context acquisition framework to support mobile spatial and context-aware applications. In *International Symposium on Web and Wire-*

*less Geographical Information Systems*, pages 100–116. Springer.

Furno, D., Loia, V., Veniero, M., Anisetti, M., Bellandi, V., Ceravolo, P., and Damiani, E. (2011). Towards an agent-based architecture for managing uncertainty in situation awareness. In *2011 IEEE Symposium on Intelligent Agent (IA)*, pages 1–6. IEEE.

Geihs, K. and Wagner, M. (2012). Context-awareness for self-adaptive applications in ubiquitous computing environments. In *International Conference on Context-Aware Systems and Applications*, pages 108–120. Springer.

Gu, T., Pung, H. K., and Zhang, D. (2007). A semantic p2p framework for building context-aware applications in multiple smart spaces. In *International Conference on Embedded and Ubiquitous Computing*, pages 553–564. Springer.

Gundersen, O. E. (2013). Situational awareness in context. In *International and Interdisciplinary Conference on Modeling and Using Context*, pages 274–287. Springer.

Guo, B., Zhang, D., and Imai, M. (2011). Toward a cooperative programming framework for context-aware applications. *Personal and ubiquitous computing*, 15(3):221–233.

Guo, B., Zhang, D., Sun, L., Yu, Z., and Zhou, X. (2013). icross: toward a scalable infrastructure for cross-domain context management. *Personal and ubiquitous computing*, 17(3):591–602.

Haque, H. M. U., Rakib, A., and Uddin, I. (2016). Modelling and reasoning about context-aware agents over heterogeneous knowledge sources. In *International conference on context-aware systems and applications*, pages 1–11. Springer.

Hegde, R. M., Kurniawan, J., and Rao, B. D. (2009). On the design and prototype implementation of a multimodal situation aware system. *IEEE transactions on multimedia*, 11(4):645–657.

Huebscher, M. C. and McCann, J. A. (2006). An adaptive middleware framework for context-aware applications. *Personal and Ubiquitous Computing*, 10(1):12–20.

Iglesias, J., Bernardos, A. M., Alvarez, A., and Sacristan, M. (2010). A light reasoning infrastructure to enable context-aware mobile applications. In *2010 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, pages 386–391. IEEE.

Indulska, J., McFadden, T., Kind, M., and Henricksen, K. (2003a). Scalable location management for context-aware systems. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 224–235. Springer.

Indulska, J., Robinson, R., Rakotonirainy, A., and Henricksen, K. (2003b). Experiences in using cc/pp in context-aware systems. In *International Conference on Mobile Data Management*, pages 247–261. Springer.

Javadzadeh, G. and Rahmani, A. M. (2020). Fog computing applications in smart cities: A systematic survey. *Wireless Networks*, 26(2):1433–1457.

Jiang, X., Chen, N. Y., Hong, J. I., Wang, K., Takayama, L., and Landay, J. A. (2004). Siren: Context-aware computing for firefighting. In *International Conference on Pervasive Computing*, pages 87–105. Springer.

Khan, L. U., Yaqoob, I., Tran, N. H., Kazmi, S. A., Dang, T. N., and Hong, C. S. (2020). Edge-computing-enabled smart cities: A comprehensive survey. *IEEE Internet of Things Journal*, 7(10):10200–10232.

Khatoun, R. and Zeadally, S. (2016). Smart cities: concepts, architectures, research opportunities. *Communications of the ACM*, 59(8):46–57.

Kim, E. and Choi, J. (2007a). A context-awareness middleware based on service-oriented architecture. In *International Conference on Ubiquitous Intelligence and Computing*, pages 953–962. Springer.

Kim, E. and Choi, J. (2007b). A semantic interoperable context infrastructure using web services. In *International Conference on Computational Science and Its Applications*, pages 839–848. Springer.

Kitchenham, B., Charters, S., Budgen, D., Brereton, P., Turner, M., Linkman, S., Jørgensen, M., Mendes, E., and Visaggio, G. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical report, Ver. 2.3, Evidence-Based Software Engineering Project, Keele University and University of Durham, UK.

Klimek, R. and Rogus, G. (2014). Modeling context-aware and agent-ready systems for the outdoor smart lighting. In *International Conference on Artificial Intelligence and Soft Computing*, pages 257–268. Springer.

Knappmeyer, M., Baker, N., Liaquat, S., and Tönjes, R. (2009). A context provisioning framework to support pervasive and ubiquitous applications. In *European Conference on Smart Sensing and Context*, pages 93–106. Springer.

Kon, F. and Santana, E. F. Z. (2016). Cidades inteligentes: Conceitos, plataformas e desafios. In Maldonado, J. C., Viterbo, J., Delamaro, M., and dos Santos Marczak, S., editors, *Jornadas de Atualização em Informática*, chapter 1, pages 13–60. Sociedade Brasileira de Computação, Porto Alegre, Brasil.

Konstantinou, N., Solidakis, E., Zafeiropoulos, A., Stathopoulos, P., and Mitrou, N. (2010). A context-aware middleware for real-time semantic enrichment of distributed multimedia metadata. *Multimedia Tools and Applications*, 46(2-3):425–461.

Lee, K. M., Kim, H.-J., Choi, K.-H., and Shin, D.-R. (2006). An intelligent middleware architecture for context-aware service discovery. In *International Conference on Computational Science*, pages 899–902. Springer.

Lee, S. (2007). Context modeling and inference system for heterogeneous context aware service. In *Symposium on Human Interface and the Management of Information*, pages 413–422. Springer.

Li, X., Eckert, M., Martinez, J.-F., and Rubio, G. (2015). Context aware middleware architectures: survey and challenges. *Sensors*, 15(8):20570–20607.

Lu, K., Nussbaum, D., and Sack, J.-R. (2007). Globecon — A scalable framework for context aware computing. In *European Conference on Smart Sensing and Context*, pages 190–206. Springer.

Machado, G. M., Maran, V., Dornelles, L. P., Gasparini, I., Thom, L. H., and de Oliveira, J. P. M. (2018). A systematic mapping on adaptive recommender approaches for ubiqui-

tous environments. *Computing*, 100(2):183–209.

Maciel, C., De Souza, P. C., Viterbo, J., Mendes, F. F., and Seghrouchni, A. E. F. (2014). A multi-agent architecture to support ubiquitous applications in smart environments. In *Agent Technology for Intelligent Mobile Services and Smart Societies*, pages 106–116. Springer.

Madhusudanan, J., Selvakumar, A., and Sudha, R. (2010). Frame work for context aware applications. In *2010 Second International conference on Computing, Communication and Networking Technologies*, pages 1–4. IEEE.

Malandrino, D., Mazzoni, F., Riboni, D., Bettini, C., Colajanni, M., and Scarano, V. (2010). Mimosa: context-aware adaptation for ubiquitous web access. *Personal and ubiquitous computing*, 14(4):301–320.

Malek, J., Laroussi, M., and Derycke, A. (2006). A middleware for adapting context to mobile and collaborative learning. In *Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)*, pages 5–pp. IEEE.

Martín, D., Lamsfus, C., and Alzua, A. (2010). Automatic context data life cycle management framework. In *5th International Conference on Pervasive Computing and Applications*, pages 330–335. IEEE.

Martin, D., Lamsfus, C., and Alzua, A. (2011). Mobile context data management framework. In *2011 Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering*, pages 73–78. IEEE.

Mitschang, B., Nicklas, D., Grossmann, M., Schwarz, T., and Hönle, N. (2005). Federating location-based data services. In *Data Management in a Connected World*, pages 17–35. Springer.

Morandi, C., Rolando, A., and Di Vita, S. (2016). *From smart city to smart region: Digital services for an Internet of Places*. Springer.

Moustaka, V., Vakali, A., and Anthopoulos, L. G. (2018). A systematic review for smart city data analytics. *ACM Computing Surveys (CSUR)*, 51(5):1–41.

Muro, M., Amoretti, M., Zanichelli, F., and Conte, G. (2012). Towards a flexible middleware for context-aware pervasive and wearable systems. *Medical & biological engineering & computing*, 50(11):1127–1136.

Nam, T. and Pardo, T. A. (2011). Smart city as urban innovation: Focusing on management, policy, and context. In *Proceedings of the 5th international conference on theory and practice of electronic governance*, pages 185–194.

Nascimento, L. V., Machado, G. M., Maran, V., and de Oliveira, J. P. M. (2021). Context recognition and ubiquitous computing in smart cities: a systematic mapping. *Computing*, 103(5):801–825.

Ngo, H. Q., Shehzad, A., Liaquat, S., Riaz, M., and Lee, S. (2004). Developing context-aware ubiquitous computing systems with a unified middleware framework. In *International Conference on Embedded and Ubiquitous Computing*, pages 672–681. Springer.

Oliveira, E. A., Koch, F., Kirley, M., and dos Passos Barros, C. V. G. (2015). Towards a middleware for context-aware health monitoring. In *International Workshop on Multiagent Foundations of Social Computing*, pages 19–30. Springer.

Pantsar-Syväniemi, S. (2011). Adaptable context-aware micro-architecture. In *International Conference on Grid and Pervasive Computing*, pages 125–132. Springer.

Park, H. and Lee, J. (2005). A framework of context-awareness for ubiquitous computing middlewares. In *Fourth Annual ACIS International Conference on Computer and Information Science (ICIS'05)*, pages 369–374. IEEE.

Park, K.-L., Kim, C.-S., Kang, C.-D., and Kim, S.-D. (2005). A programmable context interface to build a context infrastructure for worldwide smart applications. In *International Conference on Embedded and Ubiquitous Computing*, pages 795–804. Springer.

Paspallis, N. and Papadopoulos, G. A. (2014). A pluggable middleware architecture for developing context-aware mobile applications. *Personal and Ubiquitous Computing*, 18(5):1099–1116.

Perera, C., Zaslavsky, A., Christen, P., and Georgakopoulos, D. (2013). Context aware computing for the internet of things: A survey. *IEEE communications surveys & tutorials*, 16(1):414–454.

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18.

Poulcheria, B. and Costas, V. (2012). A context management architecture for m-commerce applications. *Central European Journal of Computer Science*, 2(2):87–117.

Presecan, S. and Tomai, N. (2009). Distributed context provisioning and reaction middleware. In *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*, pages 351–354. IEEE.

Pu, F., Liu, L., Shen, Y., and Cao, Q. (2010). P2p-based semantic policy infrastucture for pervasive computing. In *2010 International Conference on Computational Intelligence and Security*, pages 541–545. IEEE.

Qin, W., Shi, Y., and Suo, Y. (2007). Ontology-based context-aware middleware for smart spaces. *Tsinghua Science and Technology*, 12(6):707–713.

Qin, W., Suo, Y., and Shi, Y. (2006). Camps: A middleware for providing context-aware services for smart space. In *International Conference on Grid and Pervasive Computing*, pages 644–653. Springer.

Rahman, M. A., Hamdan, S., El Saddik, A., and Gueaieb, W. (2010). Context-aware social networks mashup: a personalized web perspective. In *2010 IEEE Instrumentation & Measurement Technology Conference Proceedings*, pages 1228–1233. IEEE.

Rahman, M. A., Kim, H.-N., El Saddik, A., and Gueaieb, W. (2014). A context-aware multimedia framework toward personal social network services. *Multimedia tools and applications*, 71(3):1717–1747.

Ranganathan, A. and Campbell, R. H. (2003). An infrastructure for context-awareness based on first order logic. *Personal and Ubiquitous Computing*, 7(6):353–364.

Rathore, M. M., Ahmad, A., Paul, A., and Rho, S. (2016). Urban planning and building smart cities based on the internet of things using big data analytics. *Computer Networks*, 101:63–80.

Riboni, D. (2019). Opportunistic pervasive computing: adaptive context recognition and interfaces. *CCF Transactions on Pervasive Computing and Interaction*, 1(2):125–139.

Riva, O. (2006). Contory: A middleware for the provisioning of context information on smart phones. In *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 219–239. Springer.

Roda, C., Navarro, E., Zdun, U., López-Jaquero, V., and Simhandl, G. (2018). Past and future of software architectures for context-aware systems: A systematic mapping study. *Journal of Systems and Software*, 146:310–355.

Roussaki, I., Strimpakou, M., and Pils, C. (2007). Distributed context retrieval and consistency control in pervasive computing. *Journal of Network and Systems Management*, 15(1):57–74.

Ryu, H., Park, I., Hyun, S. J., and Lee, D. (2007). A task decomposition scheme for context aggregation in personal smart space. In *IFIP International Workshop on Software Technolgies for Embedded and Ubiquitous Systems*, pages 20–29. Springer.

Sabagh, A. A. A. and Al-Yasiri, A. (2015). Gecaf: a framework for developing context-aware pervasive systems. *Computer Science-Research and Development*, 30(1):87–103.

Saleemi, M. M., Rodríguez, N. D., Lilius, J., and Porres, I. (2011). A framework for context-aware applications for smart spaces. In *Smart Spaces and Next Generation Wired/Wireless Networking*, pages 14–25. Springer.

Sánchez-Corcuera, R., Nuñez-Marcos, A., Sesma-Solance, J., Bilbao-Jayo, A., Mulero, R., Zulaika, U., Azkune, G., and Almeida, A. (2019). Smart cities survey: Technologies, application domains and challenges for the cities of the future. *International Journal of Distributed Sensor Networks*, 15(6):1–36.

Sashima, A., Izumi, N., and Kurumatani, K. (2003). Consorts: A multiagent architecture for service coordination in ubiquitous computing. In *International Workshop on Multi-Agents for Mass User Support*, pages 190–216. Springer.

Schmidt, H., Flerlage, F., and Hauck, F. J. (2009). A generic context service for ubiquitous environments. In *2009 IEEE International Conference on Pervasive Computing and Communications*, pages 1–6. IEEE.

Sebbak, F., Mokhtari, A., Chibani, A., and Amirat, Y. (2010). Context-aware ubiquitous framework services using jade-osgi integration framework. In *2010 International Conference on Machine and Web Intelligence*, pages 48–53. IEEE.

Shim, Y.-C. (2006). Distributed processing of context-aware authorization in ubiquitous computing environments. In *International Conference on Computational Science and Its Applications*, pages 125–134. Springer.

Silva, B. N., Khan, M., and Han, K. (2018). Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities. *Sustainable Cities and Society*, 38:697–713.

Sorici, A., Picard, G., Boissier, O., and Florea, A. (2015). Multi-agent based flexible deployment of context management in ambient intelligence applications. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pages 225–239. Springer.

Tadj, C. and Ngantchaha, G. (2006). Context handling in a pervasive computing system framework. In *Proceedings of the 3rd international conference on Mobile technology, applications & systems*, pages 13–es.

Talari, S., Shafie-Khah, M., Siano, P., Loia, V., Tommasetti, A., and Catalão, J. P. (2017). A review of smart cities based on the internet of things concept. *Energies*, 10(4):421.

Thomson, G. and Terzis, S. (2012). A middleware for pervasive situation-awareness. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 148–161. Springer.

United Nations, Department of Economic and Social Affairs - Population Division (2018). The world's cities in 2018 - data booklet. Online (link), accessed on 01-November-2021.

Van Kleek, M., Kunze, K., Partridge, K., et al. (2006). OPF: A distributed context-sensing framework for ubiquitous computing environments. In *International Symposium on Ubiquitious Computing Systems*, pages 82–97. Springer.

Xu, T., Jin, H., David, B., Chalon, R., and Zhou, Y. (2013). A context-aware middleware for interaction device deployment in ami. In *International Conference on Distributed, Ambient, and Pervasive Interactions*, pages 183–192. Springer.

Yang, S. J., Zhang, J., and Huang, A. F. (2009). Model, properties, and applications of context-aware web services. In *Machine Learning in Cyber Trust*, pages 323–358. Springer.

Yau, S. S., Huang, D., Gong, H., and Seth, S. (2004). Development and runtime support for situation-aware application software in ubiquitous computing environments. In *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, pages 452–457. IEEE.

Yılmaz, Ö. and Erdur, R. C. (2012). iConAwa - An intelligent context-aware system. *Expert Systems with Applications*, 39(3):2907–2918.

Yu, J., Huang, Y., Cao, J., and Tao, X. (2010). Middleware support for context-awareness in asynchronous pervasive computing environments. In *2010 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, pages 136–143. IEEE.

Yürür, Ö., Liu, C. H., Sheng, Z., Leung, V. C., Moreno, W., and Leung, K. K. (2014). Context-awareness for mobile sensing: A survey and future directions. *IEEE Communications Surveys & Tutorials*, 18(1):68–93.

Zheng, D., Wang, J., Han, W., Jia, Y., and Zou, P. (2006). Towards a context-aware middleware for deploying component-based applications in pervasive computing. In *2006 Fifth International Conference on Grid and Cooperative Computing (GCC'06)*, pages 454–457. IEEE.

Zhou, X., Tang, X., Yuan, X., and Chen, D. (2009). SP-BCA: Semantic pattern-based context-aware middleware. In *2009 15th International Conference on Parallel and Distributed Systems*, pages 891–895. IEEE.