Smart Regions in the Age of the Citizen Developer: A Service-Oriented Engineering Approach

Gerhard Kormann-Hainzl
IMC University of Applied Sciences Krems, Austria
gerhard.kormann@fh-krems.ac.at

Rubén Ruiz-Torrubiano*
IMC University of Applied Sciences Krems, Austria
ruben.ruiz@fh-krems.ac.at

ABSTRACT

Smart regions are mainly characterized by adopting information technology as a means for improving and enhancing management and economic development at the level of individual villages, cities and regions themselves. This enhancement can be effectively achieved by using smart service systems that focus on value co-creation among all stakeholders. In general, these services have to be engineered in a special way tailored to the particular community and use case at hand. In this paper, we propose an engineering approach centered on the concept of the citizen developer for creating, maintaining and managing those smart services. By using a domain-specific modelling language specially tailored to the smart region context, we develop a platform capable of generating smart services from a high-level description including data sources, transformations and visualizations. This results in an efficient and cost-effective development approach where the stakeholders of the smart region context themselves lead the process of building, deploying and implementing the relevant use-cases for their communities.

CCS CONCEPTS

• **Software and its engineering** → Software notations and tools; Context specific languages; Domain specific languages; • **Applied computing** → Computers in other domains; Computing in government; E-government; • **Computer systems organization** → Embedded and cyber-physical systems; Sensors and actuators.

KEYWORDS

Smart service engineering, Smart regions, Smart cities, Service-dominant logic, Citizen developer

ACM Reference Format:
Gerhard Kormann-Hainzl and Rubén Ruiz-Torrubiano. 2023. Smart Regions in the Age of the Citizen Developer: A Service-Oriented Engineering Approach. In *Central and Eastern European eDem and eGov Days 2023 (CEEeGov 2023), September 14, 15, 2023, Budapest, Hungary*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3603304.3603342

*Corresponding author

CEEeGov 2023, September 14, 15, 2023, Budapest, Hungary

ACM ISBN 979-8-4007-0006-4/23/09.
https://doi.org/10.1145/3603304.3603342

1 INTRODUCTION

Smart regions seek to enhance quality of life by adopting an effective interplay between technology and the capacity of the individuals and organizations involved to create and manage knowledge and put it into practice [1]. One key component is servitization [2], a concept that has evolved from its original conception in the general business context to other industries like manufacturing [3] and healthcare [4]. Servitization focuses on the value-adding process of economic activity in contrast to products that result in tangible goods. From the perspective of service science and service-dominant logic, value is co-created [5], which means that services are seen as a collaborative process centered on interactions based on value proposal, agreement and realization that results in mutual benefit [6]. Organizationally speaking, services are co-created in service systems, which are "configurations of people, technologies and other resources that interact with other service systems to create mutual value" [7]. Specifically, service systems can be seen as what brings a customer and a provider together to create value and benefit for all parties involved [8].

Digital technology transforms service systems into smart service systems [9]. Specifically, smartness is built into service systems by engineering intelligent software with data collection, processing and analytics capabilities to transform data into knowledge, and this knowledge into services. In general, data is generated (and sometimes also, pre-processed) by smart things (usually also referred to as smart objects, or products), which are physical products embedded with data processing and networking capabilities. In the Internet of Things (IoT) context, these smart things take the form of sensors and other devices like actuators, Radio-Frequency Identification tags (RFID) and mobile phones that are seamlessly embedded into wireless communication networks [10].

In this paper, we present a new engineering approach for smart service systems in the context of the smart region. Our approach is based on a Low-Coding/No-Coding (LCNC) solution for designing and orchestrating smart service systems that do not require traditional custom software engineering projects to come to fruition. Instead, we resort to a Domain-Specific Modeling Language (DSML) specially tailored to the smart region context. This formal language is used as the foundation for our tool. Our approach involves several advantages for the smart region: i) empowering the role of the region in designing and implementing their own use cases, ii) enhancing re-usability and knowledge sharing between communities and regions by easily adapting use cases already in use for a different community, and iii) reducing costs and time-to-market by abstracting away the main architectural properties of smart service systems and automating most of the engineering process.

This paper is organized as follows: Section 2 gives a theoretical introduction on smart service systems in the special context of the

smart region. In Section 3, we introduce our approach and elaborate on its importance from the point of view of the citizen developer. The approach is evaluated using an environmental use case analysis in Section 4. We conclude with a general discussion and remarks for future work in Section 5.

2 SMART SERVICE SYSTEMS IN THE SMART REGION CONTEXT

The term smart region is often used inconsistently or neglected in the literature on smart cities or smart villages. This paper is based on literature that describes smart regions as an independent construct that the value proposition and value creation for smartification initiatives and smart services should be tailored to specific regions and region types [11]. The different types of regions can be characterized by three variables. The first variable is the population density of a region, measured as the percentage of people living in urban or rural communities. The second variable is the distance to a populated center, also derived from the OECD's extended regional typology model. These two variables result in a classification consisting of five region types: Predominantly Urban (PU), Intermediate close to a City (INC), Intermediate Remote (INR), Predominantly Rural close to a City (PRC) and Predominantly Rural Remote (PRR). As an example, the federal state of Lower Austria includes examples of regions of all these five types. The third variable, the data ecosystem and the mechanism of co-creation of value among the stakeholders involved, was derived from the concept of data economy [11, 12].

Especially the third differentiation variable, the region-type-specific characteristics of existing data ecosystems and the framework conditions for co-creation mechanism between involved stakeholders are relevant for the model for the development of smart services presented in this paper. The value co-creation model is described in detail in Section 3.2, which on the one hand defines the regional government as an actor with overarching responsibility for the smart service infrastructure, and on the other hand distinguishes between the types of communities that are very independent and active drivers for smart services to communities that only act in a coordinating or minimalist manner and willingly reuse and continue to use the data ecosystem and existing smart services from the region.

3 A LOW-CODING ENGINEERING APPROACH FOR THE CITIZEN DEVELOPER

In general, smart services can be designed using top-down, bottom-up or middle-out approaches [13]. Top-down represents one of the most traditional approaches, where community government and central decision makers implement changes without active participation from the local communities. By contrast, in bottom-up approaches changes are initiated from the communities themselves, propagating all the way up to local governments (being implemented in practice only if approved by the local government and there is appropriate funding and technology available). In middle-out approaches, both sights are represented and meet in the middle. The main idea is to engage all stakeholders in a smart service co-design process by bringing together different backgrounds, experiences and skills.

One key inhibitor for the middle-out approach are technological hurdles, which are mostly faced by citizens that do not have an IT background [14]. It is therefore critical to lower technical entry barriers in order to achieve an effective participatory process to co-design smart services and maximize value co-creation. By empowering citizens to take a central role in the development of smart services, these are made to citizen developers [15], which can be defined as those with no proper software engineering or technical background [16]. The main tool for providing support for the citizen developers that we propose here is the use of low-code no-code (LCNC) solutions [17]. These are tools that support users in creating software by reducing the amount of code needed (low-code) or eliminating altogether the need for writing code in a formal programming language. Using LCNC solutions in the smart region context has several advantages: first, it enhances alignment between IT and the domain context by giving domain experts the possibility of developing smart services themselves. Second, it reduces costs and the need for dedicated personnel for delivering smart service projects. The citizen developer and the other stakeholders themselves can efficiently develop the required solutions without the need for lengthy explanations to IT experts or professional software developers. Additionally, LCNC approaches make it easier to quickly prototype solutions that can be refined in a later stage. For instance, citizen participation could be fostered by organizing hackathons or using gamification techniques where use cases are prototyped using no–coding solutions [18]. As an additional advantage, LCNC platforms that implement best practices in system design and security automatically enforce those practices in the produced software (for instance, using the principle of "Security by Design" [19]), increasing software quality and trust in the system.

3.1 Smart Service Lifecycle

Similar to other software engineering processes, smart services undergo the lifecycle depicted in Figure 1. We start with an initial use case analysis. In this phase, domain experts define the requirements needed to solve a given practical problem in the smart region context. For instance, this could be related to environment management (air pollution monitoring) or transport infrastructure use cases (e.g. smart parking [20]). One important aspect in this analysis are regulatory requirements that might involve usage of data or other considerations in the application domain. In the next phase (data analysis), the IoT infrastructure is evaluated and the available data is analyzed for its suitability for the use case under consideration. If the data is not suitable, a new infrastructure project needs to be started where the necessary infrastructure has to be deployed (this phase is out of the scope of this paper).

Next, the smart service is co-designed, which means that the relevant stakeholders (city and/or regional government, domain experts and citizens) participate in the process of designing the new smart service. This could include defining the functionality and UI (functional requirements) and also data flows and other requirements like security and privacy that need to be met by the new service (non-functional requirements). Citizen participation and engagement can be accomplished in different ways, e.g. by

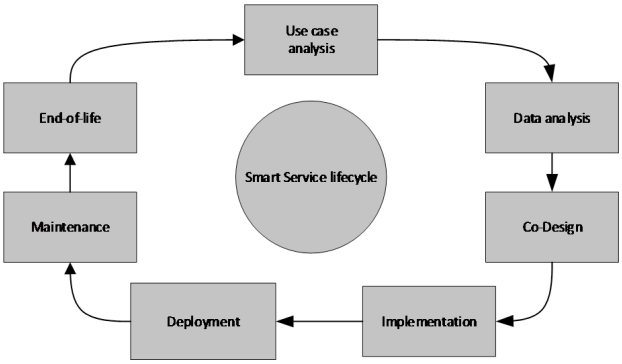

Figure 1: The smart service lifecycle (Authors' image).

organizing workshops, competitions and other participatory design activities [21].

After the co-design phase, the service is implemented using either a traditional software engineering approach or a method involving LCNC tools. When this process is finished, a software artefact (the output of the implementation phase) is rolled out in a production server, possibly in a cloud environment (a process also known as deployment). During the production time of the service, different maintenance tasks need to be performed, e.g. when the data changes or when there are new requirements or improvements that need to be done. Finally, after the service has reached its goal, it might be discontinued and archived. Note that a new use case analysis might result in the service being retrieved from the archived state (therefore the arrow from end-of-life to use case analysis in Fig. 1).

Citizen-centric participatory activities can, in principle, involve up to three different phases: the use case analysis (e.g. when citizens submit ideas and problems to be solved), co-design (when citizens themselves participate in the service design process) and implementation phases (e.g. by using LCNC solutions to actually implement the service). In the coming sections, we focus on the implementation activities.

3.2 Smart Service Value Co-Creation based on Service-Dominant Logic

The application of service-dominant (SD) logic allows a systematic development of a multi-actor perspective value-in-use based co-creation. [22-24]. As a consequence, the co-designing element and the smart service lifecycle in Figure 1 are also embedded into this multi-dimensional perspective. Methodologically, our SD model is based on the Service Dominant Business Model Radar (SMBM/R) [23]. SDBM/R is a business model development process developed for and in the context of smart mobility business models. It is a business model development process that includes five steps: 1) identifying and agreeing on the co-created value-in-use and the target customer, 2) describing the customer experience, 3) determining the components of value-in-use (the actor value proposition) and the associated actors, 4) determining the costs and benefits for each actor, and 5) determining the high-level activities that realize the value proposition for each actor. The design process is iterative and involves multiple iterations until the radar is deemed complete. The goal is to create a business model with a positive sum of costs and benefits for each actor and a positive sum of costs and benefits from a multi-dimensional perspective. Based on the high-level results, the actors integrate them into their concrete implementation and business processes.

Figure 2 conceptually outlines the Service-Dominant Smart Service Model Radar (SDSSM/R), which builds on the SDBM/R to consider the key actors, including the citizen developer as an actor in itself, for the development of smart regions specific smart services. Additionally, the SDSSM/R includes an extension to the actor co-designing level.

The defined actors are 1) the regional government, which focuses on the benefits from the perspective of the region and is responsible for the establishment and orchestration of the smart service ecosystem, such as smart service IT infrastructure and IoT platform infrastructure. On the community level, the role of the community in providing smart city infrastructure depends on the degree of data openness. Depending on the chosen approach to data openness, the city must either take responsibility for the infrastructure or share decision-making power with external parties [25].

In the model presented, the regional government has a responsibility to establish a data governance model in which communities can participate based on their data openness preferences. For this reason, we have distinguished the three categories of communities (self organizing, coordinating and minimalistic) [25] according to their willingness to actively participate in the smart service lifecycle process. The issue of data openness preferences as a distinguishing criterium is anchored at the level of the regional government. The community type A (self organizing) is very much responsible for the life cycle of smart services. Community type B (coordinating) draws heavily on resources from the entire region-specific smart services, especially for co-production, to develop their community. Community type C (minimalistic) limits itself to (re-)using existing smart services solutions, e.g. from other regions, simply because it lacks its own resources.

Among the citizens, the model distinguishes between citizen users, classic users of smart services and citizen developers, who are actively involved in the smart service life cycle as empowered users using the LCNC tools during the co-designing and co-production phases. These two phases are now separate steps in the SDSSM/R compared to the SDBM/R, which exclusively leads the actor co-production phase.

This conceptual model (SDSSM/R) is currently being developed iteratively in research projects, methodologically based on the procedure for the SSBM/R. [23]

3.3 Smart Service Engineering

In the citizen developer paradigm, the implementation of the smart service is also accomplished either in part or totally by the smart region stakeholders themselves, with a clear focus on citizen participation. In practice, this can be achieved by using LCNC tools. The activities involved are depicted in Figure 3.

We start by defining metadata, which are data about the service itself, and can be thought of as properties with assigned values. For instance, one property could be a service name or identifier,

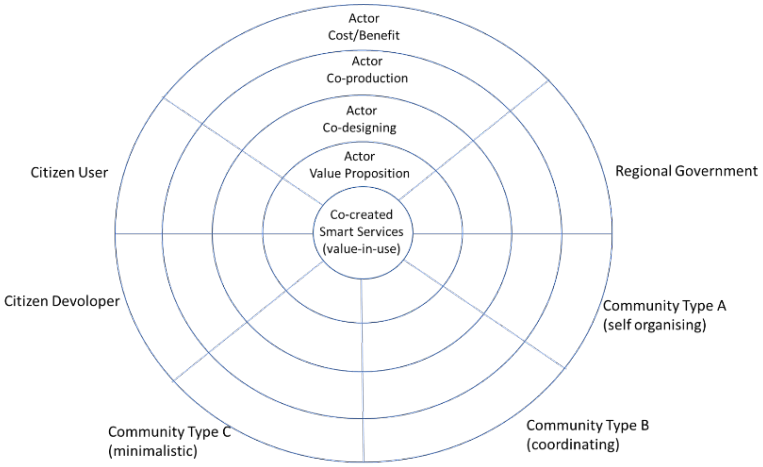

Figure 2: The Service-Dominant Smart Service Model Radar (SDSSM/R) (Authors' image).

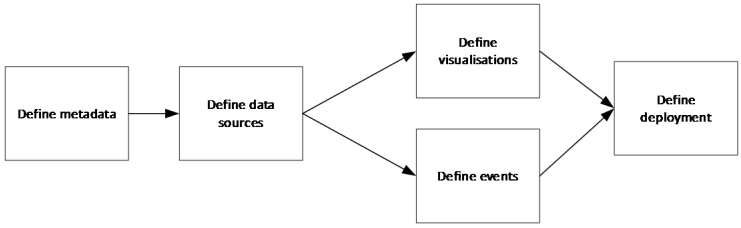

Figure 3: Smart service implementation process (Authors' image).

a version number, or the name of the author(s). One of the core activities in the implementation phase is to define the data sources. This is related to the IoT platform being used in the smart region or community under consideration. Normally, there is no direct communication between individual sensors and smart services. In contrast, data is aggregated in a middleware that is usually called broker or gateway. As an example, consider the Orion Context Broker that is at the edge of the Fiware IoT platform. The smart service is attached to the broker by means of an application programming interface (API). Therefore, once that the IoT platform is known, the LCNC solution can just use a predefined library to fetch data from the broker programmatically. The user only needs to define, in principle, which endpoints (i.e. unique identifiers or addresses) are necessary to interact with the broker and how data needs to be prepared or filtered. For example, in the case of environmental data, the stakeholders might be interested in ozone levels in different locations, but not in CO2 or NOX, so these other data are filtered out.

Next, we need to define how to present the data to the user (visualizations). Here, we first need to define which type of application best suits the use-case at hand. Usually, web applications are used since they can be accessed from anywhere using a web browser, possibly complemented by a smartphone app. Concrete visualizations like maps, charts and diagrams are defined in this step as well. Additionally, events like push notifications can be defined to alert users if e.g. some values are exceeded depending on the use-case. Finally, details of the deployment need also be specified, like domain names, IP addresses and containerization technology.

Once these definitions are finished, a smart service can be generated that fulfils the requirements. We propose a layered architecture for smart services as shown in Figure 4. At the bottom, the IoT layer is responsible for the direct communication with the IoT middleware. In parallel, the data access layer represents the interface to a database management system (DBMS) that is used to store users, preferences and settings, among other necessary data access objects. Both layers provide data to be used by the data processing layer. This is one of the core layers of the smart service where data is transformed, filtered and prepared for visualization. Typical operations to be performed at the level of this layer are missing data imputations, grouping selection of attributes/features and tabular dataset formation. Once the data is in the appropriate form, the visualization layer provides the necessary mechanisms and frameworks to realize concrete data visualizations, like street maps, line and pie charts, or other types of diagrams and tables. Finally, the user interface layer provides a unified user experience integrating all visual and interaction elements using an appropriate technology, like web frontend frameworks or mobile apps.

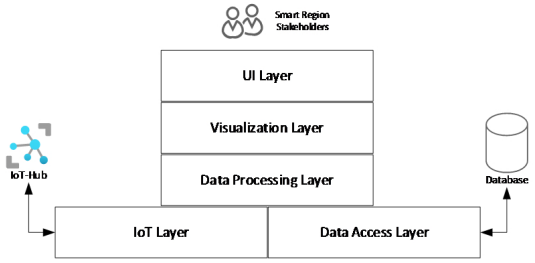

Figure 4: General architecture of a smart service (Authors' image).

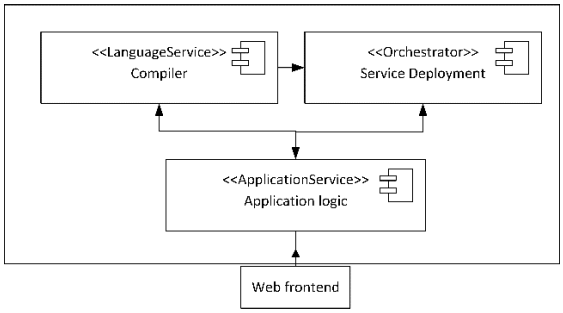

Figure 5: Component diagram for LCNC tool (Authors' image).

3.4 Low-Coding Smart Services

We now introduce a general architecture for a LCNC tool for smart service engineering and a sample implementation. The tool currently under development is called Sagittarius[1] and implements the process depicted in Figure 3 and its main goal is to produce smart services that satisfy the architecture outlined in Figure 4. In Figure 5 we show the reference architecture that we used to implement our tool as a UML component diagram.

The tool is composed of several components in a service-oriented architecture. The ApplicationService is a central gateway acting as a controller for the web frontend. The web frontend contains an editor and additional functionality for compiling and deploying smart services. The LanguageService compiles the smart service definition into a ready-to-deploy application. We chose to implement the domain-specific modelling language described in [26] due to its flexibility and its proximity to natural language. This service is also responsible for adding the necessary code to connect the service to a specific IoT platform, like Fiware. Finally, the Orchestrator is in charge of deploying, starting, stopping smart services in a coordinated way. For this purpose, a containerization technology like Docker is used to deploy the service in a public or private cloud environment.

4 USE CASE EVALUATION: ENVIRONMENTAL DATA ANALYSIS

In this section, we present a concrete use case that was implemented using the proposed LCNC tool. We took air pollution measurements from the city of Madrid (which is a PR region according to the classification presented in Section 2) collected in a publicly available Fiware instance as data source. We choose this example because the data is publicly available to showcase our LCNC tool in a concrete use case, but the tool can in principle be used in any other region where the corresponding infrastructure is available. In this instance, several measurement stations distributed over the city are used to collect data regarding ozone, carbon dioxide and nitrogen oxide levels. The use case is air quality monitoring: the principal requirement of the service is to show air quality measurements in a publicly accessible dashboard. The main workflow in the tool is shown in Figure 6 (top). The central part of the tool is the code editor in the middle panel, which is accompanied by a left panel showing a tree-like structure. This structure is used for classifying smart services in different categories like water management, environment, transport, etc. For instance, if the user selects the item "water management", they would see all smart services that were classified in this category (similar to files in a folder). In Step 1, we use the editor for implementing the process described in Figure 2 using the domain-specific modelling language [26]. This step can be divided into three sub-steps. In Step 1a, we define general metadata for the service like name and version. In Step 1b, the data source and the relevant fields are specified (in this example, the data fields of interest are NOX, ozone, the location of the measurement station, and the date of the observation). In Step 1c, visualizations are defined. In this case, we specify a map visualization where points in the map show levels of NOX and ozone. Additionally, we would like to have a detailed view on a particular measurement station over time. To achieve this, we group by location and plot NOX and ozone levels as a function of time. To conclude, details about the deployment are specified in the last part (not shown).

In Step 2, the service specification is compiled - i.e. the service is translated into an application written in a high-level programming language. For our tool, we chose Python as the implementation language and Plotly as a visualization framework. In Step 3 (bottom part of Figure 6), the service is deployed as a stand-alone application and can be used by the users in the community.

There are different ways in which the workflow described above can be used in service co-design activities. In the following, we give some examples:

- The tool could be used as a demonstrator of a quick prototype by domain experts to get a first approximation to the problem to be solved.
- It could also be used as a tool in an appathon-like event where citizens themselves compete with their prototypes in the context of a given use-case.
- Additionally, the tool could be used in workshops where the involved stakeholders (city administration and management) participate in the design process.
- In self-organizing communities, the tool can be used within the IT-department to implement the service after receiving specifications from domain experts in natural language form.
- Engaged individual citizens can use the tool to submit proposals for new services as definitions or prototypes to city management.

[1]Source code available at https://github.com/IMC-UAS-Krems/Sagittarius

Figure 6: Creating an air pollution monitoring service with an LCNC tool.

5 DISCUSSION AND FUTURE WORK

In this paper, a new framework for smart service engineering based on co-design activities and supported by a LCNC solution was presented. The framework encourages citizen participation in smart service design processes by setting a common standard and a central tool supporting the co-design, implementation and deployment phases of the smart service lifecycle. In this way, we effectively propose to decentralize and open this process to all stakeholders involved. This results in benefits like increased citizen engagement, more efficient design processes and cost savings. Moreover, the experience of one community can be easily shared with other communities in the same or different regions (by sharing smart service definitions), paving the way for a global smart service co-design community that shares use-cases and the knowledge to implement them.

The next steps in the development of this tool involve adding more visualizations and event-driven capabilities like e.g. receiving notifications in case of measurements exceeding a specified threshold. Additionally, predictive analytics using machine learning will be included to support decision making and to gain insights from the data in a transparent way.

We note that, although the formal language used was designed to be as near to natural language as possible, it still might represent an entry barrier for users without IT experience. Therefore, the next step involves a special user-interaction layer to enable users to write the service specification directly in natural language by using e.g. generative AI language methods. Additionally, a concrete implementation roadmap for all stakeholders as well as a concise investigation of the legal aspects involved will be developed as part of future work.

REFERENCES

[1] M. Markkula and H. Kune, "Making Smart Regions Smarter: Smart Specialization and the Role of Universities in Regional Innovation Ecosystems," Technology Innovation Management Review, vol. 5, no. 10, pp. 7–15, 2015.
[2] S. Vandermerwe and J. Rada, "Servitization of business: Adding value by adding services," European Management Journal, vol. 6, no. 4, pp. 314–324, Dec. 1988, doi: 10.1016/0263-2373(88)90033-3.
[3] T. S. Baines, H. W. Lightfoot, O. Benedettini, and J. M. Kay, "The servitization of manufacturing: A review of literature and reflection on future challenges," Journal of Manufacturing Technology Management, vol. 20, no. 5, pp. 547–567,

Jan. 2009, doi: 10.1108/17410380910960984.
[4] J. Zhang and L. Qi, "Crisis Preparedness of Healthcare Manufacturing Firms during the COVID-19 Outbreak: Digitalization and Servitization," International Journal of Environmental Research and Public Health, vol. 18, no. 10, Art. no. 10, Jan. 2021, doi: 10.3390/ijerph18105456.
[5] R. F. Lusch, S. L. Vargo, and G. Wessels, "Toward a conceptual foundation for service science: Contributions from service-dominant logic," IBM Systems Journal, vol. 47, no. 1, pp. 5–14, 2008, doi: 10.1147/sj.471.0005.
[6] S. L. Vargo and R. F. Lusch, "Service-dominant logic: continuing the evolution," J. of the Acad. Mark. Sci., vol. 36, no. 1, pp. 1–10, Mar. 2008, doi: 10.1007/s11747-007-0069-6.
[7] P. P. Maglio, S. L. Vargo, N. Caswell, and J. Spohrer, "The service system is the basic abstraction of service science," Inf Syst E-Bus Manage, vol. 7, no. 4, pp. 395–406, Sep. 2009, doi: 10.1007/s10257-008-0105-1.
[8] D. Beverungen, O. Müller, M. Matzner, J. Mendling, and J. vom Brocke, "Conceptualizing smart service systems," Electronic Markets, vol. 29, no. 1, pp. 7–18, Mar. 2019, doi: 10.1007/s12525-017-0270-5.
[9] D. Beverungen, C. F. Breidbach, J. Poeppelbuss, and V. K. Tuunainen, "Smart service systems: An interdisciplinary perspective," Information Systems Journal, vol. 29, no. 6, pp. 1201–1206, Nov. 2019, doi: 10.1111/isj.12275.
[10] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," Computer Networks, vol. 54, no. 15, pp. 2787–2805, Oct. 2010, doi: 10.1016/j.comnet.2010.05.010.
[11] G. Kormann-Hainzl, H. Lovasz-Bukvova, and M. Hoezl, "Are smarts villages just smaller smart cities? Call for a region-type-specific approach to the smartification of communities," presented at the Central and Eastern European e/Dem and e/Gov days 2021, 2021.
[12] M. Brezzi, L. Dijkstra, and V. Ruiz, "OECD Extended Regional Typology: The Economic Performance of Remote Rural Regions," OECD, Paris, Aug. 2011. doi: 10.1787/5kg6z83tw7f4-en.
[13] J. Fredericks, G. A. Caldwell, and M. Tomitsch, "Middle-out design: collaborative community engagement in urban HCI," in Proceedings of the 28th Australian Conference on Computer-Human Interaction, in OzCHI '16. New York, NY, USA: Association for Computing Machinery, Nov. 2016, pp. 200–204. doi: 10.1145/3010915.3010997.
[14] A. Wolff, M. Barker, L. Hudson, and A. Seffah, "Supporting smart citizens: Design templates for co-designing data-intensive technologies," Cities, vol. 101, p. 102695, Jun. 2020, doi: 10.1016/j.cities.2020.102695.
[15] M. Oltrogge *et al.*, "The Rise of the Citizen Developer: Assessing the Security Impact of Online App Generators," in 2018 IEEE Symposium on Security and Privacy (SP), May 2018, pp. 634–647. doi: 10.1109/SP.2018.00005.
[16] "Quickbase: How Companies Are Coding More Apps With Fewer Developers | Fortune." https://fortune.com/2016/08/30/quickbase-coding-apps-developers/ (accessed Mar. 20, 2023).
[17] F. Sufi, "Algorithms in Low-Code-No-Code for Research Applications: A Practical Review," Algorithms, vol. 16, no. 2, Art. no. 2, Feb. 2023, doi: 10.3390/a16020108.
[18] A. Wolff *et al.*, "Engaging with the Smart City Through Urban Data Games," in Playable Cities: The City as a Digital Playground, A. Nijholt, Ed., in Gaming Media and Social Effects. Singapore: Springer, 2017, pp. 47–66. doi: 10.1007/978-981-10-1962-3_3.
[19] C. Zolotas, K. C. Chatzidimitriou, and A. L. Symeonidis, "RESTsec: a low-code platform for generating secure by design enterprise services," Enterprise Information Systems, vol. 12, no. 8–9, pp. 1007–1033, Oct. 2018, doi: 10.1080/17517575.2018.1462403.
[20] T. Lin, H. Rivano, and F. Le Mouël, "A Survey of Smart Parking Solutions," IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 12, pp. 3229–3253, Dec. 2017, doi: 10.1109/TITS.2017.2685143.
[21] D. Gooch *et al.*, "Amplifying Quiet Voices: Challenges and Opportunities for Participatory Design at an Urban Scale," ACM Trans. Comput.-Hum. Interact., vol. 25, no. 1, p. 2:1-2:34, Jan. 2018, doi: 10.1145/3139398.
[22] J. Yu, Y. Wen, J. Jin, and Y. Zhang, "Towards a service-dominant platform for public value co-creation in a smart city: Evidence from two metropolitan cities in China," Technological Forecasting and Social Change, vol. 142, pp. 168–182, May 2019, doi: 10.1016/j.techfore.2018.11.017.
[23] O. Turetken, P. Grefen, R. Gilsing, and O. E. Adali, "Service-Dominant Business Model Design for Digital Innovation in Smart Mobility," Bus Inf Syst Eng, vol. 61, no. 1, pp. 9–29, Feb. 2019, doi: 10.1007/s12599-018-0565-x.
[24] P. Ekman, J. Röndell, and Y. Yang, "Exploring smart cities and market transformations from a service-dominant logic perspective," Sustainable Cities and Society, vol. 51, p. 101731, Nov. 2019, doi: 10.1016/j.scs.2019.101731.
[25] M. Buchinger, P. Kuhn, and D. Balta, "Towards Interoperability of Data Platforms for Smart Cities," in Handbook of Smart Cities, J. C. Augusto, Ed., Cham: Springer International Publishing, 2020, pp. 1–22. doi: 10.1007/978-3-030-15145-4_70-1.
[26] Ruiz-Torrubiano, Rubén, Dhungana, Deepak, Kormann-Hainzl, Gerhard, and Paudel, Sarita, "SSDL: A Domain-Specific Modeling Language for Smart City Services," in To be published in Smart Services Summit 2022, Zürich: Springer.