Essential Means for Urban Computing

Specification of Web-Based Computing Platforms for Urban Planning, a Hitchhiker's Guide

Abstract

This paper provides an overview of the specifications of web-based computing platforms for urban data analytics and computational urban planning practice. There are currently a variety of tools and platforms that can be used in urban computing practices, including scientific computing languages, interactive web languages, data sharing platforms and still many desktop computing environments, e.g. GIS software applications. For an urban planner, however, working with most of these platforms often entails dealing with many technicalities such as data format conversions and establishing connections in between these platforms, let alone working with different interfaces and programming languages. We have reviewed a list of promising technologies considering their ease of use and applicability in urban planning and urban data analytics. This review is not only based on the technical factors such as capabilities of the programming languages and database management systems but also the human factors of the computing technologies. We have considered human factors related to three groups of human users, namely urban planners without structured knowledge of computing, research and design software developers, and enthusiastic citizens. The arena of web-based computing platforms is currently under rapid development and too volatile to be predictable, therefore, in this paper we focus on the specification of the requirements and potentials from an urban planning point of view rather than speculating about the fate of computing platforms or programming languages. The paper concludes with a list of most promising computing technologies and a technical specification of an ideal urban computing platform.

Keywords

Urban Computing, Spatial Computation, Planning Support, Decision Support, Dataflow Programming

1. Introduction

In this paper we focus on the applications of urban computing in Smart Cities Planning practice (M. Batty et al., 2012).

1.1. What is Urban Computing?

It is difficult to provide a comprehensive definition of the emerging field of Urban Computing (a.k.a. Urban Informatics, especially in the context of social informatics). In short urban computing is a process of acquisition, integration, and analysis of (big) data generated by diverse sources such as sensing technologies and large-scale computing infrastructures in the context of urban spaces, the volume, velocity and variety of such data often requires the use of cloud computing infrastructure and software services (Hashem et al., 2015). Urban Computing aims to improve urban planning & decision-making, e.g. to reduce traffic congestion or energy consumption (Zheng, Capra, Wolfson, & Yang, 2014) & (Foth, Choi, & Satchell, 2011). Urban Computing is applicable in a variety of fields, namely:

- environmental modelling (e.g.(Shang, Zheng, Tong, Chang, & Yu, 2014; Zheng, Liu, & Hsieh, 2013)),
- modelling energy use/generation (e.g. (Simão, Densham, & (Muki) Haklay, 2009)),
- transport modelling (e.g. (Zheng, Liu, Yuan, & Xie, 2011)),
- monitoring health (e.g. (Varshney, 2007)),
- epidemiology (e.g. (Lopez, Gunasekaran, Murugan, Kaur, & Abbas, 2015)),
- social informatics (e.g. (Foth, Forlano, Satchell, Gibbs, & Donath, 2011), & (Pires & Crooks, 2017)),
- criminology (e.g. (Bogomolov et al., 2014)),
- participatory planning (e.g. (Robinson & Johnson, 2016; Tenney & Sieber, 2016)).

1.2. Why is Urban Computing needed in Urban Planning?

In Urban Planning, we are often interested in analysing the so-called what-if scenarios using simulations and projections (Michael Batty & Torrens, 2001). Traditionally, the geo-spatial analysis of intervention scenarios, urban plans, and urban data is done by means of the so-called Geographic Information Systems (GIS) and Planning Support Systems (PSS) [see (Harris & Batty, 1993) & (Michael Batty, 2007)] and Spatial Decision Support Systems SDSS); where the term system implies the use of a database, a kernel/library of computational methods for geo-spatial data processing, and an interface. Despite the technical similarities in using a spatial database, and so forth, the two categories are different in that the SDSS are geared towards ***operational*** decision-making whereas the PSS are geared towards ***strategic*** planning, which often involves land-use planning and therefore considering the land-use transport interactions (the distinction between PSS & SDSS from (Geertman & Stillwell, 2009)). As the name system suggests, these systems are often closed-sourced desktop applications based on SQL database management systems (ibid). In that sense, urban computing, deemed as spatial analysis of urban data is nothing new. However, the prospect of urban computing is based on the new horizons provided by the web-based computing platforms for the so-called operational, managerial, and strategic planning actions (Couclelis, 2005):

Past - - - - - - - → ***Present*** - - - - - - - → ***Future***

React, Respond, Mitigate, Control, Manage, Adapt, Anticipate, Prepare, Change, Shape, Create

Operational - - - - - - - → ***Managerial*** - - - - - - - → ***Strategic***

Figure 1: The Temporal Spectrum of Planning Actions, reproduced from (Couclelis, 2005)

While most of the scholarly work in the area of PSS is focused on land-use change, there are other aspects of urban dynamics that could be modelled computationally, that is to say the broader discussion is on what changes can be explained, anticipated, and taken into account when making strategic decisions on spatial plans, this broader field of research and development is called Urban Modelling (Michael Batty, 2009). Considering the nature of outcomes of planning processes, (e.g. land-use plans) we can conclude that the spatial relations between land-use distributions and a variety of phenomena need to be considered in making strategic planning decisions, such things as land-use & transport interactions and their effects on energy use in transport (see (Keirstead, Jennings, & Sivakumar, 2012)) and the effect of land-use distribution on bio-diversity and the use of natural resources (especially water) should ideally be considered when proposing plans. From a pragmatic point of view, however, the adoption of PSS in practice is not high (Geertman & Stillwell, 2009):

> *"It is disturbing, in fact, to observe the extent to which new computer-based support systems are developed by researchers to the point of adoption but are never implemented in planning practice or policy making. Similarly, there is evidence to indicate that systems which are made operational are not extensively used, after the initial novelty has passed, by those planning organizations for which they have been developed in the first instance. In terms of application, it is possible to point to more failures than successes, i.e. to more cases where systems have not been implemented than examples where they are used routinely. Moreover, many state- of-the-art systems appear to take a long time to reach the 'market' and this is often a process requiring considerable financial resources(ibid)."*

We suggest that the reserch and development culture of Spatial Planning and Decision Support Systems (SPDSS, terminology of (Geertman & Stillwell, 2009)) must adopt **open-source** & **agile** development principles for effective 'market' uptake and ensuring the viability of the R&D products (Approach, 2009; Crowston & Howison, 2005; Hey & Payne, 2015; von Krogh, 2003). In short, what it takes to go in this direction is primarily the availability of suitable platforms for rapid prototyping, development, release, sharing, and test of SPDSS (incorporating a variety of Urban [Analysis/Simulation] Models).

1.1. Problem Statement

We do not need to focus on the Graphical User Interface of proprietary or open-source desktop applications. If we change our focus from the interface to the essential means of geo-spatial computing, and consider that for inter-disciplinary computational research we need to make use of a variety of other kinds of mathematical libraries, we see that a desktop system cannot be the answer to such needs. There are two problems with using the currently available GIS desktop applications for innovative inter-disciplinary research in Urban Computing applied in Urban Planning (i.e. Design and Development of Web-Based Planning Support Systems):

- Data-Related Problems:
  - Data-Availability: how easy is it to acquire a relevant dataset?
  - Data-Interoperability: how easy is it to read/write datasets from/to file formats?
  - Data-Mergeability: how easy is it to overlay multiple datasets?
- Workflow-Related Problems:
  - Workflow Comprehensibility: to what extent is the whole workflow understandable?
  - Workflow Editability: how easy is it to modify the workflow explicitly?
  - Workflow Repeatability: how easy is to repeat a certain data processing workflow?
  - Workflow Shareability: how easy is it to share a workflow from one system to another?
  - Workflow Scalability: how easy is to process large datasets with a workflow?
  - Workflow Sustainability: to what extent is the workflow modular and recyclable?

A less discoursed matter about the planning support systems is the very social/human process of developing them. These systems can be developed by Research Software Engineers[1]. A typical research software developer is not necessarily a software engineer, but usually a domain-specific researcher who can develop software or computational workflows. A typical research software engineer, often does not have the means of a software vendor to develop a large application and a graphical user interface. The core of the work of research software development is on developing analytic workflows[2].

2. What do we need for urban computing?

It is well known that the time spent on research and development is often much more valuable than the computation time. Therefore, to answer the above question, not only we need to consider the technical specification of a computing platform but also the human interface requirements with regards to the ease of ideation-development-test cycles (prototyping).

2.1. Visual Dataflow Programming

In processing big data, there are generally to approaches, namely batch processing and real-time processing (Hashem et al., 2015). Considering the real-time data processing requirement, especially in dealing with managerial and operational planning actions (see Figure 1), we can conclude that the Dataflow Programming[3] is an appropriate paradigm for setting up an R&D/prototyping environment (Blackstock & Lea, 2014) & (Szydlo, Brzoza-Woch, Sendorek, Windak, & Gniady, 2017). Considering that the sustainability, repeatability of the workflow, it is practical to adopt a modularization & standardization approach to workflow development. Standardization is important for reusability. Particularly, when thinking of the intermediate steps of a workflow system: if each "block" of a workflow supports inputs and produces outputs on a standard format, then other tools can be produce / consume the inputs / outputs of a process. Of course, having a visual overview of the workflow is of high added value, as it makes the workflow as intuitive as a flowchart. The idea of a visual dataflow programming language is to represent the high-level logic of a program/workflow as a graph of nodes, which are blocks of (reusable/shareable) code. The representation of the high-level logic as a graph makes it easy to focus on the complex big-picture for a group of developers working on a workflow. Instead of developing a complete software application with a graphical user interface, a research software engineer can focus on the core of the workflow, model the workflow, test it, share it, and release it as a functional prototype.

If the workflow description language is a (de facto) standard, the intended user does not need to learn a new interface to interact with the workflow. In other words, instead of focusing on optimizing a new software application in terms of its interface and the computational efficiency, more attention can be paid to the effectivity of the workflow itself.

In short, adopting a visual cloud-based dataflow processing language (and ecosystem) brings about a few advantages:

- Automation of repetitive tasks for data cleansing, validation, etc.
- Informal and yet sustainable standardization based on common-practices and bottom-up emergence of workflow patterns[4]
- Sharing workflow pattern solutions instead of re-inventing the wheel
- The possibility of interdisciplinary collaboration
- Ultimate modularization of workflows based on sharing nodes/blocks of code
- Agile development-test-release cycles
- Promotion of Open-Source development practices and therefore rapid progress
- Ensuring re-usability and repeatability of workflow based practices such as spatial analyses
- Saving time and resources by significantly reducing the time and effort in re-inventing interfaces
- Raising the level of comprehensibility of analytic workflows by providing a glass-box view of the process (as opposed to black-box SPDSS)
- The possibility of public participation in planning processes by means of rapid development and integration of apps (e.g. using Node-RED).

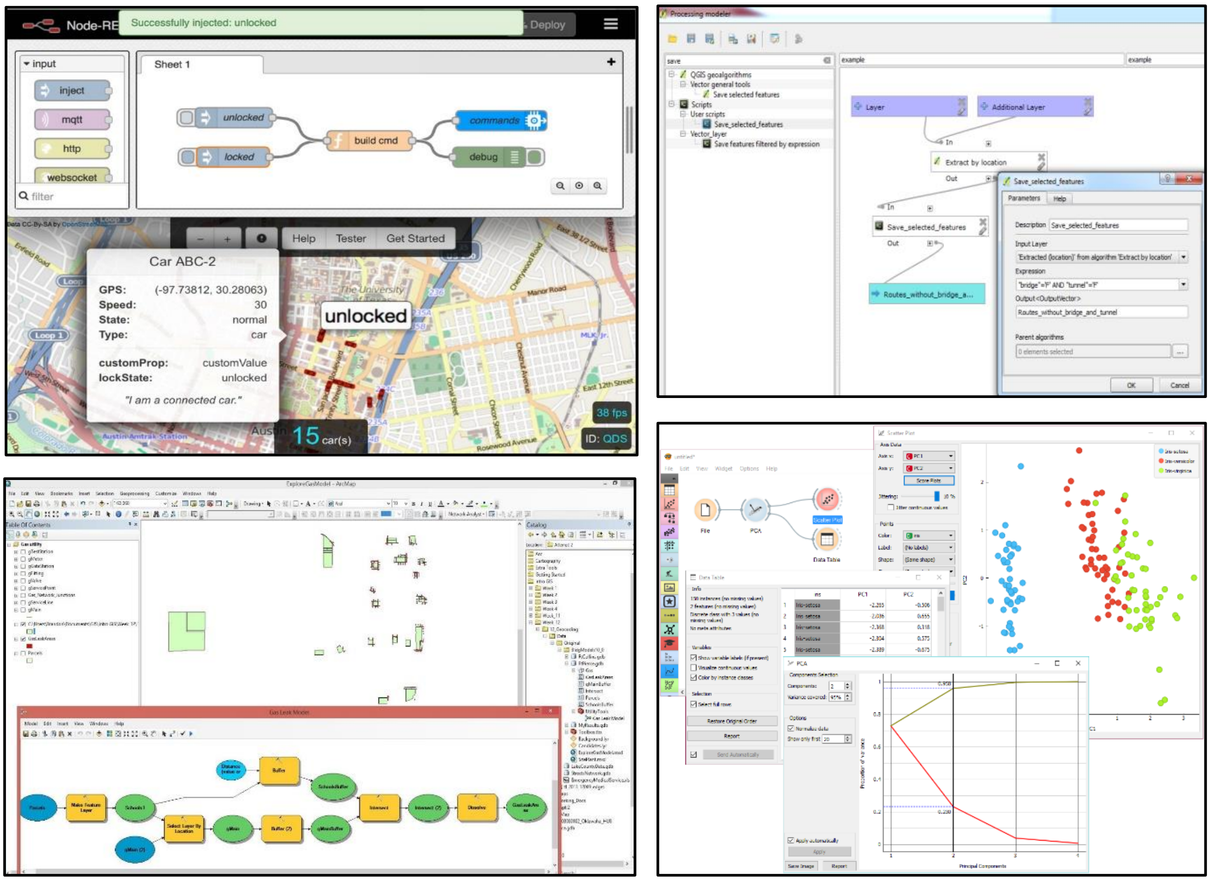

Figure 2: data processing workflow examples, respectively from top left, clockwise, node-RED[5] (picture from (Boyd, 2015)*), QGIS Graphical Modeller[6], Anaconda Orange[7], and ArcGIS Model Builder[8]*

2.1. Spatial Computing Modules

Here we provide an overview of the requirements of a software application for urban computing; and focus on the specific functionalities that deal with geo-spatial data. Geo-spatial data can be analysed in at least four spatial forms from most concrete to most abstract:

1. *Geographical Data Models:* with geographically positioned (points, lines, polygons, polyhedrons)
2. *Geometrical Data Models:* with points, lines, polygons, polyhedrons (in local coordinate systems)
3. *Topological Data Models:* with vertices, edges, faces, bodies (algebraic topology)
4. *Graphical Data Models:* with objects, links (Graph Theory)
5. *Spectral Data Models:* with eigenvectors, eigenvalues

The use of the last category of data models is relatively newer than the other types of the models and is used in for modelling the dynamics of diffusion flows and Markov Processes in networks (Volchenkov & Blanchard, 2007), , (Wei & Yao, 2014), (P. Nourian, Rezvani, Sariyildiz, & van der Hoeven, 2016), (Pirouz Nourian, 2016). Performing spectral analyses requires using a computational linear algebra library such as NumPy[9]. In Table 1, we have categorized the specifically required functionalities for spatial computing as to the previously introduced fields of application of urban computing.

Table 1: a list of typical goals, required spatial data types, and typical analytic (mathematical) or simulation (computational) modelling approaches of urban computing

	Goal	**Typically *Required Spatial Data Types***	***3D?***	***Typical Modelling Methodologies***
[Land-Use &] Transport Modelling	understanding *potentials* (accessibility) and predicting the *dynamics* of mobility [& land-use change]	road network lines, land-polygons, cellular phone network data, GPS trajectories, etc.	possibly beneficial	Discrete-Choice Modelling, Gravity Models, Agent-Based Modelling (ABM), Cellular Automata (CA), Markov Chains, Operations Research
Sociometrics & Econometrics	understanding *potentials*, and *dynamics* of social and economic interactions	demographic data attributed to building, block, district, city, region polygons, crowd-sourced geo-tagged data points, etc.	probably unnecessary	Markov Chains, Markov Chain Monte Carlo (MCMC), Network Centrality, Artificial Intelligence, Statistical Modelling, Predictive Analytics
Criminology & Crime Prevention	understanding potentials, and dynamics of crime in cities	road-networks, demographics attributed to building, & city polygons, geo-tagged (positioned) spatial crime data, etc.	possibly beneficial	Statistical Modelling, Predictive Analytics, Agent-Based Modelling (ABM), Cellular Automata (CA), Markov Chains, Monte Carlo Simulation
Energy Modelling	understanding *potentials*, and *dynamics* of energy use and [renewable] energy generation	3D polyhedral models of buildings, point clouds	necessary	Solar Irradiance Simulation (requiring geometric intersections), Computational Fluid Dynamics (CFD, recurring raster and vector fields and differential operators), Monte Carlo Methods
Environmental Modelling	understanding *potentials*, and *dynamics* of environmental threats & opportunities (air pollution, noise, vegetation, etc.)	aerial photos, point clouds, vector maps, raster maps	necessary	Analytic Models and Simulation Models (e.g. CA and ABM), Complex System Dynamics, Hydrology, Complex Adaptive Systems

In Figure 3 we have shown the inter-relations between the data-models and methods (processes) of spatial computing.

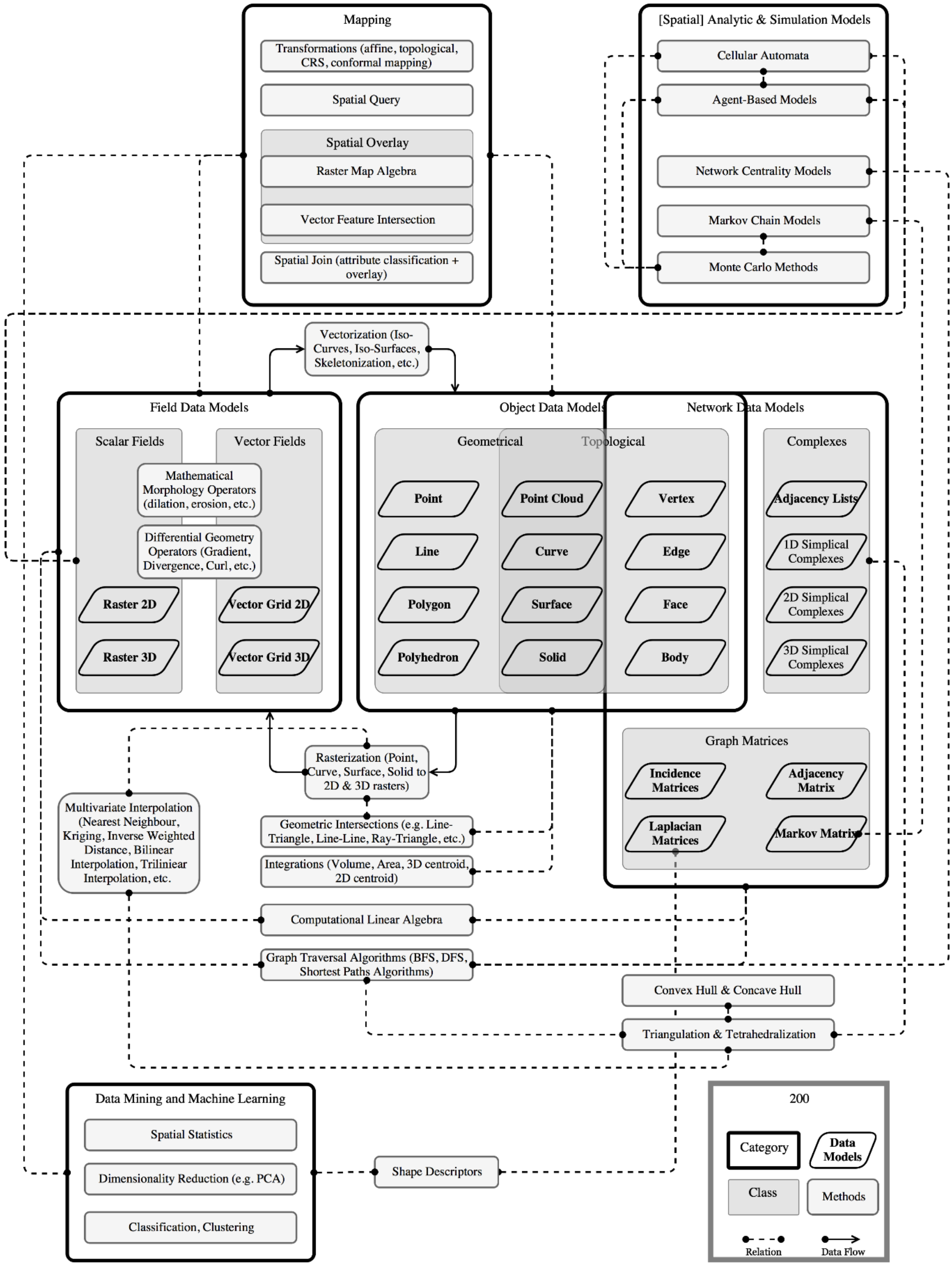

Figure 3: typically required data models and processes in urban computing

3. Promising Technologies for Urban Computing

We have identified a few promising technologies for urban computing, based on Python, Java and JavaScript ecosystems. In this overview we consider their potential for prototyping, geo-spatial mapping, 3D visualization, handling big data, and numerical computing. We also stress that these technologies are not mutually exclusive, but can (in some cases) be used in combination.

Java: for example in a web-GIS for environmental analyses has been prototyped by (Zavala-Romero et al., 2014) presents a flexible platform for making dynamic maps using OpenLayers[10]. The FIWARE platform (Zahariadis et al., 2014) offers an "Application MashUp Generic Enabler", i.e. the WireCloud[11] for visual programming and prototyping web applications. Another flow-based programming environment for Java development supported by Apache Hadoop[12] is NiFi[13]. Java can also provide for interactivity and 3D visualization. The OpenGeoSpatial foundation (aka OSGeo[14]) also provides an open source GIS toolkit for Java called GeoTools[15]. Considering the might of Hadoop for big data analytics and the support of OSGeo Java seems to be a fertile language for urban computing. One option for 3D visualization in Java is JogAmp[16].

Python: for example the Geoda-Web[17], that is the web-based version of CAST[18] with its spatial analysis library PySal[19] seems to be a promising open-source project. Python is the de facto language of open-source development in the field of Geo information science, e.g. as available in QGIS. Python provides a wide range of libraries for numerical and scientific computing such as NumPy, SciPy and Pandas, which facilitates development. Interactive development environments such as IPython (Interactive Python) (Perez & Granger, 2007) and web-based Jupyter notebooks (Shen, 2014) seems to be the most promising technology for prototyping and interactive computing. Some universities have started facilitating the use of Jupyter interactive documents as a common means of exchanging reproducible research products, e.g. on JupyterHub[20], NBViewer[21], or SURF-sara (Templon & Bot, 2016) provide hosting and viewing services for sharing Jupyter notebooks. A few options which stand out for 3D visualization in Python are: MatPlotLib[22], Mayavi[23] or VisPy[24].

JavaScript: for example, the Carto[25] SaaS (Software as a Service, formerly known as CartoDB[26]) provides user-friendly Web-GIS tools. However, it does not support any visual form of explicit workflow development. The other promising JavaScript platform for spatial analysis is MapBox[27], which offers access to the Turf library[28]. Node-RED (Blackstock & Lea, 2014), based on IBM BlueMix (a.k.a. IBM Cloud)[29], seems to be the most promising technology in terms of visual programming and the ease of prototyping Internet of Things (IoT) applications. In addition, interactive visualization in web-browsers is well supported, and arguably more advanced than comparable libraries in Python by JavaScript, thanks to the D3.js library, by Mike Bostock[30] (Bostock, Ogievetsky, & Heer, 2011). In addition to D3 for interactive graphics, there is three.js[31] for WebGL rendering in the browser. Other JavaScript libraries which should not go unnoticed for urban computing are Leaflet[32] (mobile-friendly interactive maps providing access to OSM[33]) and Cesium[34], the latter providing for quality 3D visualization.

As mentioned above, each of these directions (Java, Python, or JavaScript) comes with their own advantages and shortcomings. It is difficult (and perhaps futile) to point to one of these languages as the most promising language for urban computing. However, it can be said that each of them is stronger in a certain direction, respectively: Java in handling big data, Python in (mathematical) research workflows, and JavaScript in IoT. Figure 4 shows a summary of the potentials described above and the usage of a hypothetical urban computing platform for five types of users, namely citizens, developers, researchers, IoT technology providers, and computing platform service providers.

In addition, it is noteworthy to mention that in the related field of computer-aided design (CAD), there is an active movement towards development of visual programming languages and connecting them together by means of a cloud platform, Flux[35], initially sponsored by Google[36]. Considering the attractiveness of aligning urban design and urban planning actions, it would be ideal to work in an environment where planners, designers, and research software engineers could all work and share their workflows, for example, a 3D city modelling SaaS such as Tygron[37] or CityZenith[38] could potentially become such a shared development environment.

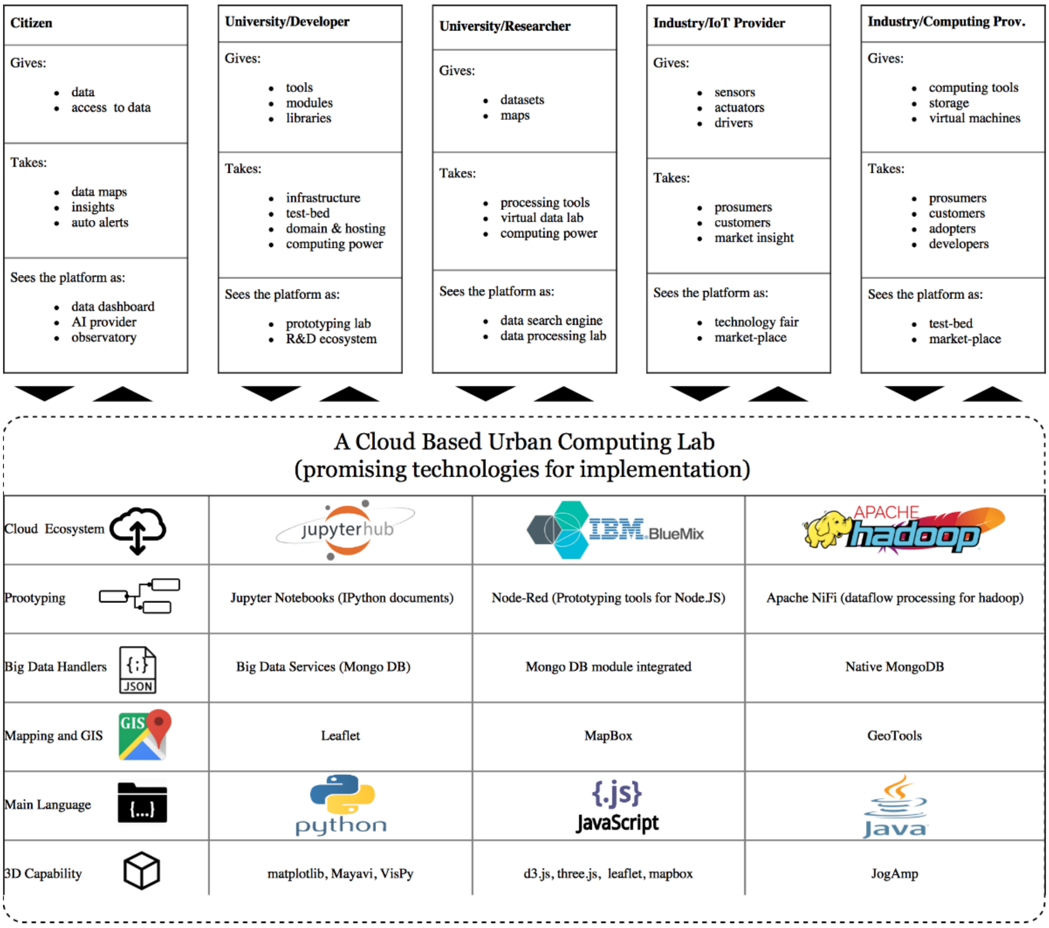

Figure 4: the use-cases of an urban computing platform and the technologies that provide for making such a platform

4. Conclusion

In response to this question: "What are the essential means for urban computing?", we have provided an overview of specific functionalities required in dealing with geo-spatial data processing (spatial analysis), referred to as spatial computing in Table 1, showed the interrelations between spatial data models and main categories of methods for spatial computing in Figure 3, which we deem as the essential means for urban computing. We have considered three programming languages and their promising aspects for urban computing. In Figure 4 we show a summary of the comparison between these languages and their related platforms as to their potential usage in developing modern Spatial Decision Support Workflows (SDSW). Each of the languages is strong in one direction, therefore, depending on the specific need for R&D for developing a SDSW any of them can be considered the best language.

5. References

Approach. (2009). The Manifesto for Agile Software Development, *7*(2). Retrieved from http://nlp.chonbuk.ac.kr/SE/ch05.pdf

Batty, M. (2007). Planning support systems: progress, predictions, and speculations on the shape of things to come. *Planning Support Systems for Urban and Regional Analysis*, *44*(0), 0–18. http://doi.org/10.1103/PhysRevE.78.016110

Batty, M. (2009). Urban Modelling - Chapter XXX. *International Encyclopedia of Human Geography*, 1–18. Retrieved from www.elsevierdirect.com/brochures/hugy/overview.html

Batty, M., Axhausen, K. W., Giannotti, F., Pozdnoukhov, A., Bazzani, A., Wachowicz, M., … Portugali, Y. (2012). Smart cities of the future. *European Physical Journal: Special Topics*, *214*(1), 481–518. http://doi.org/10.1140/epjst/e2012-01703-3

Batty, M., & Torrens, P. M. (2001). Modelling complexity : The limits to prediction. *Cybergeo*. http://doi.org/10.4000/cybergeo.1035

Blackstock, M., & Lea, R. (2014). Toward a Distributed Data Flow Platform for the Web of Things (Distributed Node-RED). *Proc. of the 5th International Workshop on Web of Things*, 34–39. http://doi.org/10.1145/2684432.2684439

Bogomolov, A., Lepri, B., Staiano, J., Oliver, N., Pianesi, F., & Pentland, A. (2014). Once Upon a Crime: Towards Crime Prediction from Demographics and Mobile Data. In *ICMI'14* (pp. 427–434). Istanbul: ACM. http://doi.org/10.1145/2663204.2663254

Bostock, M., Ogievetsky, V., & Heer, J. (2011). D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, *17*(12), 2301–2309. http://doi.org/10.1109/TVCG.2011.185

Boyd, B. (2015). Build a connected-car IoT app with Geospatial Analytics. Retrieved November 19, 2017, from https://www.ibm.com/developerworks/library/mo-connectedcar-app/index.html

Couclelis, H. (2005). "Where has the future gone?" Rethinking the role of integrated land-use models in spatial planning. *Environment and Planning A*, *37*(8), 1353–1371. http://doi.org/10.1068/a3785

Crowston, K., & Howison, J. (2005). The social structure of free and open source software development. *First Monday*, *2*(SPEC). http://doi.org/10.5210/fm.v10i2.1207

Foth, M., Choi, J. H., & Satchell, C. (2011). Urban informatics. *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work - CSCW '11*, 1. http://doi.org/10.1145/1958824.1958826

Foth, M., Forlano, L., Satchell, C., Gibbs, M., & Donath, J. (2011). *From Social Butterfly to Engaged Citizen: Urban Informatics, Social Media ... - Marcus Foth, Laura Forlano, Christine Satchell, Martin Gibbs, Judith Donath - Google Books*. MIT Press. Retrieved from https://books.google.ca/books?hl=en&lr=&id=EdLxCwAAQBAJ&oi=fnd&pg=PR5&dq=theory,+social+media,+civic+engagement&ots=jttOa3qDhD&sig=RXMu4Y9Fox2wXPQjzpI6kRs7AUk#v=onepage&q=theory%2C social media%2C civic engagement&f=false

Geertman, S., & Stillwell, J. C. H. (2009). *Planning Support Systems Best Practice and New Methods*. Retrieved from http://books.google.es/books?id=2KSzOr1UULoC

Harris, B., & Batty, M. (1993). Locational Models, Geographic Information and Planning Support Systems. *Journal of Planning Education and Research*, *12*(3), 184–198. http://doi.org/10.1177/0739456X9301200302

Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Ullah Khan, S. (2015). The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, *47*, 98–115. http://doi.org/10.1016/j.is.2014.07.006

Hey, T., & Payne, M. C. (2015). Open science decoded. *Nature Publishing Group*, *11*. http://doi.org/10.1038/nphys3313

Keirstead, J., Jennings, M., & Sivakumar, A. (2012). A review of urban energy system models: Approaches, challenges and opportunities. *Renewable and Sustainable Energy Reviews*, *16*(6), 3847–3866. http://doi.org/10.1016/j.rser.2012.02.047

Lopez, D., Gunasekaran, M., Murugan, B. S., Kaur, H., & Abbas, K. M. (2015). Spatial big data analytics of influenza epidemic in Vellore, India. In *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014* (pp. 19–24). IEEE. http://doi.org/10.1109/BigData.2014.7004422

Nourian, P. (2016). Configraphics: Graph Theoretical Methods for Design and Analysis of Spatial Configurations. *Doi.Org*. http://doi.org/10.7480/abe.2016.14.1348

Nourian, P., Rezvani, S., Sariyildiz, I. S., & van der Hoeven, F. D. (2016). Spectral Modelling for Spatial Network Analysis. *Proceedings of the Symposium on Simulation for Architecture and Urban Design (simAUD 2016)*. Retrieved from https://repository.tudelft.nl/islandora/object/uuid:81c02b9c-3ddc-4273-8c2b-7e84c6dc7604

Perez, F., & Granger, B. E. (2007). IPython: A system for interactive scientific computing. *Computing in Science and Engineering*, *9*(3), 21–29. http://doi.org/10.1109/MCSE.2007.53

Pires, B., & Crooks, A. T. (2017). Modeling the emergence of riots: A geosimulation approach. *Computers, Environment and Urban Systems*, *61*, 66–80. http://doi.org/10.1016/j.compenvurbsys.2016.09.003

Robinson, P. J., & Johnson, P. A. (2016). *Civic Hackathons: New Terrain for Local Government-Citizen Interaction? Urban Planning* (Vol. 1). http://doi.org/10.17645/up.v1i2.627

Shang, J., Zheng, Y., Tong, W., Chang, E., & Yu, Y. (2014). Inferring gas consumption and pollution emission of vehicles throughout a city. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '14*, 1027–1036. http://doi.org/10.1145/2623330.2623653

Shen, H. (2014). Interactive notebooks: Sharing the code. *Nature*, *515*(7525), 151–152. http://doi.org/10.1038/515151a

Simão, A., Densham, P. J., & (Muki) Haklay, M. (2009). Web-based GIS for collaborative planning and public participation: An application to the strategic planning of wind farm sites. *Journal of Environmental Management*, *90*(6), 2027–2040. http://doi.org/10.1016/j.jenvman.2007.08.032

Szydlo, T., Brzoza-Woch, R., Sendorek, J., Windak, M., & Gniady, C. (2017). Flow-Based Programming for IoT Leveraging Fog Computing. In *2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)* (pp. 74–79). IEEE. http://doi.org/10.1109/WETICE.2017.17

Templon, J., & Bot, J. (2016). The Dutch National e-Infrastructure, 13–18. Retrieved from http://pos.sissa.it/

Tenney, M., & Sieber, R. (2016). *Data-Driven Participation: Algorithms, Cities, Citizens, and Corporate Control*. *Urban Planning* (Vol. 1). http://doi.org/10.17645/up.v1i2.645

Varshney, U. (2007). Pervasive healthcare and wireless health monitoring. *Mobile Networks and Applications*, *12*(2–3), 113–127. http://doi.org/10.1007/s11036-007-0017-1

Volchenkov, D., & Blanchard, P. (2007). Random walks along the streets and canals in compact cities: Spectral analysis, dynamical modularity, information, and statistical mechanics. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, *75*(2). http://doi.org/10.1103/PhysRevE.75.026104

von Krogh, G. (2003). Open-source software. *MIT Sloan Management Review*, *44*(3), 14–18. Retrieved from https://en.wikipedia.org/wiki/Open-source_software#Open-source_software_licensing

Wei, X., & Yao, X. A. (2014). The random walk value for Ranking spatial characteristics in road networks. *Geographical Analysis*, *46*(4), 411–434. http://doi.org/10.1111/gean.12064

Zahariadis, T., Papadakis, A., Alvarez, F., Gonzalez, J., Lopez, F., Facca, F., & Al-Hazmi, Y. (2014). FIWARE Lab: Managing Resources and Services in a Cloud Federation Supporting Future Internet Applications. In *2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing* (pp. 792–799). IEEE. http://doi.org/10.1109/UCC.2014.129

Zavala-Romero, O., Ahmed, A., Chassignet, E. P., Zavala-Hidalgo, J., Fernández Eguiarte, A., & Meyer-Baese, A. (2014). An open source Java web application to build self-contained web GIS sites. *Environmental Modelling and Software*, *62*, 210–220. http://doi.org/10.1016/j.envsoft.2014.08.029

Zheng, Y., Capra, L., Wolfson, O., & Yang, H. (2014). Urban Computing. *ACM Transactions on Intelligent Systems and Technology*, *5*(3), 1–55. http://doi.org/10.1145/2629592

Zheng, Y., Liu, F., & Hsieh, H.-P. (2013). U-Air. *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '13*, 1436. http://doi.org/10.1145/2487575.2488188

Zheng, Y., Liu, Y., Yuan, J., & Xie, X. (2011). Urban computing with taxicabs. *Proceedings of the 13th International Conference on Ubiquitous Computing - UbiComp '11*, 89. http://doi.org/10.1145/2030112.2030126

[1] http://rse.ac.uk/who/
[2] http://www.commonwl.org/
[3] https://stackoverflow.com/questions/461796/dataflow-programming-languages/2035582
[4] http://www.workflowpatterns.com/
[5] https://nodered.org/
[6] https://docs.qgis.org/2.8/en/docs/user_manual/processing/modeler.html?highlight=workflow
[7] https://orange.biolab.si/screenshots/
[8] http://pro.arcgis.com/en/pro-app/help/analysis/geoprocessing/modelbuilder/what-is-modelbuilder-.htm
[9] http://www.numpy.org/
[10] http://openlayers.org/
[11] https://catalogue.fiware.org/enablers/application-mashup-wirecloud
[12] http://hadoop.apache.org/
[13] https://hortonworks.com/apache/nifi/
[14] http://www.osgeo.org/
[15] http://www.geotools.org/
[16] http://jogamp.org/
[17] http://spatial.uchicago.edu/geoda-web
[18] https://geodacenter.github.io/CAST/
[19] http://pysal.readthedocs.io/en/latest/users/tutorials/dynamics.html
[20] https://github.com/jupyterhub
[21] https://nbviewer.jupyter.org/
[22] https://matplotlib.org/index.html
[23] http://docs.enthought.com/mayavi/mayavi/
[24] http://vispy.org/index.html
[25] https://carto.com/blog/how-to-use-spatial-analysis-in-your-site-planning-process/
[26] https://cartodb.github.io/training/intermediate/columbia-sipa.html
[27] https://www.mapbox.com/help/how-analysis-works/
[28] http://turfjs.org/
[29] https://www.ibm.com/cloud/
[30] https://bl.ocks.org/mbostock
[31] https://threejs.org/
[32] http://leafletjs.com/

[33] http://www.openstreetmap.org

[34] https://cesiumjs.org

[35] https://flux.io/

[36] https://bimandintegrateddesign.com/2014/10/24/googles-bim-busting-app-for-design-and-construction/

[37] http://www.tygron.com/

[38] http://www.cityzenith.com/smartworld