Technical Specification Double-side Page

1. **TECHNICAL SCOPE:** Summarize the mock-up devised during the EXPLORE phase: how have you addressed the challenge/Theme Challenges and tackled with its requirements and data. Include a diagram.

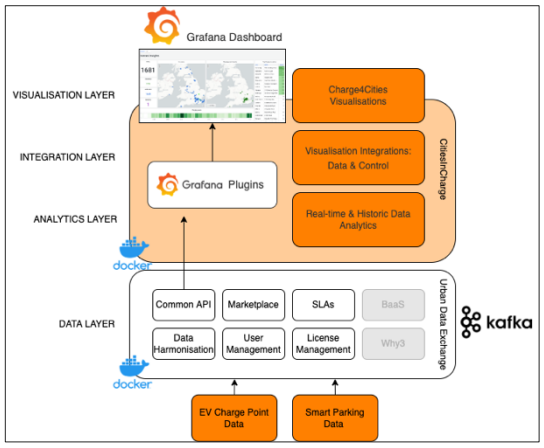
CitiesInCharge is a data-driven decision support tool for local authorities to better manage their public EV charging infrastructure and maximise its utilisation. The proposed solution uses real-time data from EV charge points, smart parking sensors and other open data sources. These data streams are onboarded and harmonised in our existing Urban Data Exchange (UDX) product which then powers the decision-making tools on top.

Through a series of workshops and user interviews we have identified the key challenges for local authorities as:

1) Understanding demand and forecasting future demand to support future roll out of further EV charge points
2) Understanding infrastructure health to minimise downtime and enforce SLAs with charge point operators
3) Identifying violations in charge point usage for better enforcement. E.g., non-charging vehicles blocking chargers
4) Understanding energy demands, current revenue, and energy pricing
5) Having a consolidated view across multiple providers of charge point infrastructure in the local authority

Our technical mock-up is a **fully functional prototype** based on live data from EV charging bays that provides a preview of how challenges 1, 2 and 5 can be addressed. Challenges 3 and 4 are outlined in design mock-ups (illustrated in technical pitch slides). The diagram below shows the architecture of our mock-up and the technologies involved.

Real-time data streams from 3 charge point operators have been onboarded to UDX, plus live data from smart parking sensors. As the data flows through UDX, it is harmonised to common data formats (adopting the Smart Data Models for EV charging stations and parking spaces) and served to the CitiesInCharge layer via a common API.



We will extend UDX with Why3 and BaaS tools from the REACH toolkit for trusted data sharing of real-time data and smart contract management in later stages.

In the CitiesInCharge layer, three functional blocks for analytics, integrations and visualisations are emulated using Grafana and Grafana Plugins for rapid prototyping.

The Infinity plugin for Grafana interacts with the UDX common API. Harmonised timeseries data is loaded into the plugin, processed and filtered to drive the live Grafana dashboards on top. When moving to the next stage, we intend to utilise Jupyter Notebook in this layer for rapid prototyping of ML and complex analytics during development.

In the prototype demo, two dashboards provide insights on charge point demand to support decision making on future roll-out (challenge 1 above) and charge point health to support decision making on infrastructure management (challenge 2 above). Dashboard controls allow users to filter dashboard views by charge point provider (challenge 5).

2. **ALGORITHMS, TOOLS AND CONCLUSIONS:** Detail the algorithms and tools identified to accomplish the challenge/Theme Challenges. Show clear understanding of the used REACH dataset/s and addressed challenge/Theme Challenges.

We are utilising 3rd party data from our own providers, including timeseries data from EV charge points and smart parking sensors. When moving to the next stage, we will add open data on new vehicle registrations and land use.

For **Data Harmonisation** we map proprietary data to Smart Data Models (SDM), an increasingly adopted data standard by cities. There are 600+ SDMs available describing all types of data including parking and EV charging events. To perform harmonisation, we are making use of a npm package called 'object-mapper' which we have extended to fit our needs. As the real-time data passes through UDX we apply a transformation object (developed bespoke) using the object-mapper to transform the inbound payload into a standardised outbound JSON payload. The transformation objects map one data type to a specific SDM. We have developed a transformation object that maps EV charging data (typically in OCPI format) to the EVChargingStation SDM and another to map smart parking data to the ParkingSpace SDM. The transformation objects are added to our growing library (internal IP) making data integration in different environments easier over time.

For **Demand Forecasting** we plan to utilise a type of linear regression model. Based on current explorations, time series regression techniques such as Autoregression and Vector Auto Regression (VAR) show good alignment to our use case. VAR is particularly suited for demand forecasting on Multivariate Time Series where there is more than one time-dependent variable, and each variable depends not only on its past values but also has some dependency on other variables. In our case we want to forecast the quantity of future charging events based past events and other

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 951981

predictive variables such as new electric vehicle registrations.

Jupyter Notebook with Python libraries such as Numpy, SciPy and Scikit-learn will be used to prototype and develop ML algorithms further. For production deployment, ML/AI cloud services such as AWS Sagemaker will be utilised.

Data Integration and Event Inference. In some use cases it is necessary to combine multiple timeseries data to detect events of interest. For example, we want to provide insights on charge point violations such as a vehicle blocking a charging bay while not charging. To do this we combine EV charging event streams with data from parking occupancy sensors installed at charging bays. Data will be integrated by location and time and then correlations in charging status and parking status will be used to infer whether a particular charge point is blocked.

Jupyter Notebook with libraries such as Pandas will be used to prototype and develop data integration and inference algorithms further. For production deployment this process will be deployed as a Dockerised microservice.

3. **SCALABILITY AND FLEXIBILITY OF THE SOLUTION:** Discuss whether the solution can truly cope with humongous and increasing datasets and how flexible it is to adapt to other related domains

We use highly scalable industry-standard tools for big data (e.g., Kafka), as well as containerisation and virtualisation. All can be scaled horizontally and vertically as data volumes increase. Additional nodes can be added to our hosted Kafka cluster or databases for horizontal scaling, and node sizes can be increased for vertical scaling. Multiple instances of microservices are deployed to ensure continuous service provision and additional microservice instances can be immediately deployed to handle increases in data and web traffic.

By building on top of UDX we can easily plug-in data from other local authorities and use existing transformations to very quickly harmonise and serve this new data to drive decision-making tools for them. Indeed, CitiesInCharge is one use case that can build on top of UDX but we imagine customers will have similar data-related challenges in other areas that can be addressed in a similar way with the same technology stack.

Additionally, our architecture decouples the data processing components and the visualisation layer allowing us to build on top of flexible dashboarding tools such as Grafana, Quicksight, Power BI, etc. This enables us to leverage common visualisation features and only extend with specific features where necessary.

4. **DATA GOVERNANCE AND LEGAL COMPLIANCE:** Describe the security level of the proposed solution, i.e. how authentication, authorization policies, encryption or other approaches are used to keep data secure. Explain how will be compliant with the current data legislations concerning security and privacy (e.g. GDPR).

For security and privacy we also adopt industry standard tools and techniques. For user authentication we utilise AWS Cognito. API access for data providers is managed through API tokens which can be rotated as required. Hosted cloud infrastructure such as Kafka clusters and databases benefit from cloud provider security and are configured in line with well-architected best practices. All data is encrypted in transit and at rest.

The data streams contain sensor data or other infrastructure and event data that should not contain any personally identifiable or sensitive information. We specify this to data providers in our service T&Cs. However, application data does contain user information such as name, email address, etc. To comply with GDPR we strictly follow guidelines and requirements on collecting, storing and using user data. Our privacy policy has complete disclosure of personal data collection and intended use. UDX offers the ability to specify data licenses or bespoke sharing agreements on a per data stream level to handle commercial sensitivities of EV charge point providers regarding use of their data.

5. **QUALITY ASSURANCE AND RISK MANAGEMENT:** Describe the quality process planed for the final product. Technologically, which are the potential risks in all the phases of the project (design of the solution, development, testing, deployment…) and indicate mitigation plans to still fulfil the challenge/Theme Challenges and data provider requirements.

Design risks include developing a product that does not meet user needs. For example, the analytics and insights tools could deliver the wrong information or deliver information in the wrong format that does not support easy decision making or action. To mitigate this risk, we have already closely engaged with potential end users of CitiesInCharge at Surrey county council. We are also building a picture of typical organisational structures and decision-making processes through interviews with other local authorities to clarify who the individual users of our tools will be and how they fit into existing business processes to ensure our service can be adopted into business-as-usual with minimal barriers.

During development phases of the proposed solution, we will continue to adopt a rapid-prototyping and co-design approach, using tools such as Jupyter Notebook and Grafana so we can deliver and iterate quickly, fail fast, and gain useful feedback from end users.

We will use version control systems for code tracking and utilise CI/CD pipelines to support continuous integration and deployment with automated unit tests, integration testing and E2E testing. For ML algorithms this will also include testing of model fitting. All cloud infrastructure will be managed via Terraform scripts following an Infrastructure-as-code approach, so infrastructure provisioning and configuration is also tracked via version control systems.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 951981