Technical Specification Double-side Page

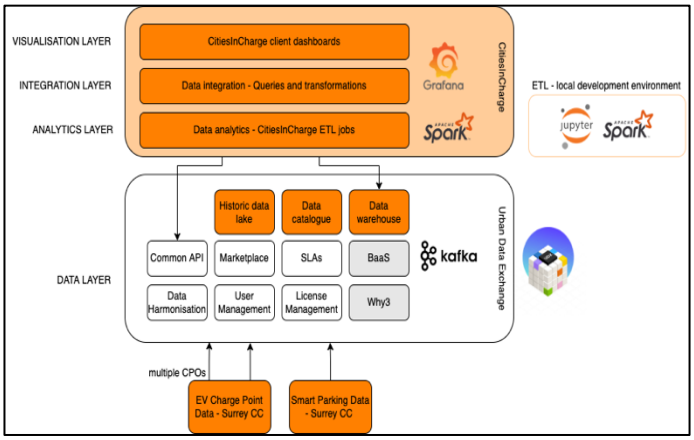
1. **TECHNICAL SCOPE:** Summarize the solution developed during the EXPERIMENT phase: how have you finally addressed the challenge/Theme Challenges and tackled with its requirements and data. Include a diagram.

CitiesInCharge is a data-driven decision support tool for local authorities to better manage their public EV charging infrastructure and its future rollout, while maximising its utilisation. Our solution enables a new data value chain by leveraging real-time data from EV charge points of different charge point providers and data from smart city infrastructures such as smart parking sensors. These multi-stakeholder data streams are securely onboarded and harmonised in our existing Urban Data Exchange (UDX) product, so they can be exchanged in an interoperable and trusted way for further data value creation. Through further pre-processing of the data streams, storage, analysis, and visualisation, we realise our CitiesInCharge decision support tool on top of these streams.

The EXPERIMENT phase allowed us to refine requirements and co-design prioritised features for CitiesInCharge in close collaboration with our data provider (and customer) Surrey County Council as follows:

1) Provide a consolidated view across multiple providers of charge point infrastructure in the local authority
2) Understand infrastructure health to minimise downtime and enforce SLAs with charge point operators
3) Understand energy consumption, CO2 emissions, and CO2 savings across the network for reporting
4) Understand network usage and running costs of the charging network
5) Identify violations in charge point usage for better enforcement. E.g., non-charging vehicles blocking chargers

The diagram below shows the solution architecture, with additions developed during REACH highlighted with orange.



Real-time data from multiple charge point operators plus smart parking sensors are onboarded to UDX and harmonised. However, to address the requirements above, a combination of real-time data and historic data is necessary so that current activities as well as historic trends can be presented in CitiesInCharge.

As such, we extended the UDX platform to handle large volumes of historic data by adding a data lake, data catalogue and data warehouse. Real-time data from UDX streams is now also piped into the data lake as well as the existing time-series database.

Crawlers frequently parse the data lake to keep the catalogue up to date as the lake grows. CitiesInCharge ETL jobs then operate on the data lake and catalogue to extract required insights and store them in the warehouse ready to drive tool dashboards. Our production ETL jobs run in the cloud, while a replica environment for local development using Jupyter Notebook and Spark allows us to massively reduce deployment and feedback times when building and testing jobs before cloud deployment.

The decision support tool is developed on top of the Grafana open-source platform. Query and transformation languages including SQL and Jsonata are used to query the warehouse for historic data and the UDX common API for real-time data. Additional plugins are utilised to integrate with the warehouse and API as data sources, and to also visualise the data as required.

2. **ALGORITHMS, TOOLS AND CONCLUSIONS:** Detail the algorithms and tools finally selected to accomplish the challenge/Theme Challenges. Summarize the main results that you have obtained during the EXPERIMENT phase: data, insights, conclusions and the main contributions to solve the challenge/Theme Challenges.

The **data lake** is an AWS S3 bucket due to S3's ability to store huge volumes of unstructured data at low cost. Data in the lake is partitioned by UDX stream ID and date using Hive style partitioning. To pipe data into the lake we use a Kafka Sink Connector for S3 from our existing Kafka cluster. The **data catalogue** is held in AWS Glue and contains a record of the partitions in the lake and their schema. Glue Crawlers are used to crawl the lake every hour to discover any new partitions and update the catalogue accordingly. The **data warehouse** is currently a Postgres instance running in AWS RDS which can be easily scaled, however, when significantly larger volumes of scale are required, the warehouse can be migrated to a more dedicated service such as AWS Redshift. For **visualisations**, we initially considered AWS Quicksight, but opted for **Grafana** due to its superior real-time data support.

The ETL jobs are Python scripts using PySpark which run on top of Apache Spark. In the cloud they are deployed and managed through AWS Glue Jobs. We develop these jobs locally in a Jupyter Notebook running on top of Spark which mimics the AWS Glue Job environment. There are currently 3 ETL jobs in operation:

Charging point count runs monthly and calculates the number of charging points that were online for the last month. This enables insights into how the network is growing over time.

Charging point status runs daily and calculates daily statistics based on status updates from the chargers, including the percentage of time that each charger has been a) operational and b) charging throughout the day. This enables

insights into the health and usage of each charger over time and whether maintenance and utilisation KPIs are met.

Charging session statistics runs monthly and calculates daily statistics from all the charging sessions that occurred in the previous month, including min/max/mean/total values for a) session durations, b) revenue and c) energy consumption. This enables insights into energy demands, revenue generation, potential revenue loss, network operational costs and CO2 emissions and savings.

From historic data analysis we have already helped our customer identify a) their network is operational about 90% of the time (in line with SLAs), b) the network is currently less than 10% utilised, c) the total revenue from charge point users is not fully covering their energy costs, d) they are potentially losing £4k+ revenue each month due to low utilisation or outages, e) erroneous data comes from operator systems and must be filtered out to not bias metrics.

Real-time data processing of the charge point data event stream provides operational decision-making support and includes live charging network health and use, and infrastructure usage violations obtained from a fusion with occupancy data from smart parking sensors.

We also explored a **demand-forecasting algorithm** to aid decision making for future rollout, based on Vector Auto Regression techniques but held back on its implementation as we addressed higher priority features for our customer.

3. **SCALABILITY AND FLEXIBILITY OF THE SOLUTION:** Explain how the solution copes with the challenge/Theme Challenges requirements and how can it be adapted to other similar problems. What work is still pending to create a real/stable product if any? What TRL level is it in?

By building on top of UDX we can easily plug-in harmonised EV data from other local authorities to drive the CitiesInCharge for their charging network. CitiesInCharge is one use case that is built on top of UDX but other data-related challenges for new customers can be addressed in a similar way with the same technology stack. For new use cases we can easily onboard required data to UDX, harmonise the data, set up necessary ETL pipelines and create additional visualisations in Grafana to meet specific requirements and insight needs for that new use case.

In terms of technical scalability, we use industry-standard tools for big data as well as containerisation and virtualisation. All aspects of the UDX platform and CitiesInCharge can be scaled horizontally and vertically as data volumes and user traffic increase. Pending work includes ongoing monitoring and optimisation of ETL jobs to reduce runtimes as data increases. Our core UDX is at TRL8 while the newly developed CitiesInCharge tool is at TRL7.

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

For user authentication to UDX we utilise AWS Cognito. CitiesInCharge currently uses a separate Authentication process, but we plan to implement SSO using UDX credentials in the near future. All access to data streams is managed through API tokens which can be rotated as required. Data licenses or bespoke sharing agreements can also be assigned on a per data stream level to handle commercial sensitivities of EV charge point providers regarding use of their data. We are fully cloud hosted and all our cloud infrastructure benefits from cloud provider security configured in line with well-architected best practices. All data is encrypted in transit and at rest.

The data streams in UDX and CitiesInCharge 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.

5. **QUALITY ASSURANCE AND RISK MANAGEMENT:** Describe the quality process followed for the final product. Technologically, which problems have you encountered and how you have solved them, and any processes followed that guarantee that the solution fulfills the challenge/Theme Challenges and data provider requirements.

To ensure we remained aligned with customer requirements we followed a co-design process with regular bi-weekly meetings at the end of each development sprint where the latest tool updates were demoed and feedback gathered. However, during early sprints most development was backend engineering to support historic data management which is not particularly visible or relevant to our customer. To grow engagement and momentum during this engineering-heavy period we therefore developed visualisations initially based on mock data to illustrate current design thinking and gain valuable feedback before gradually replacing mock data with real data and insights.

Developing CitiesInCharge introduced a significant number of new concepts, tools and ecosystems. Upskilling was required around data formats and schemas (Avro, Parquet), Kafka connect, AWS Glue and crawlers, Spark, PySpark, and Glue ETL Job environments. Developers took courses, attended REACH seminars and used pair programming for knowledge transfer.

A significant challenge was developing ETL jobs for AWS Glue in a timely manner. Doing so entirely in the Glue cloud environment was not feasible due to extremely long deployment and feedback times. Our solution was to develop our own local development environment that mimics the Glue environment using Jupyter and Spark. This reduced ETL development times by orders of weeks and can be reused for all ETL job development beyond CitiesInCharge.

All code developed for CitiesInCharge and UDX is covered by a substantial unit and integration test framework, including for all ETL jobs. Testing is integral to our CICD pipelines. Additionally, all cloud infrastructure is managed as Terraform code and under version control.

Annex 1. Means for accessing the MVP

Please, indicate in 1 page indicating the means for accessing the MVP for a potential customer (login information, website address, link to a demo video or whatever means are needed to check that the MVP exists and works).

The CitiesInCharge solution is available at:
https://ev-analytics.urbandata-dev.exchange/

Login details for demo user are:
Username: demo-user
Password: demo123