Software Architecture Design of Cloud Platforms in Automotive Domain: An Online Survey

Ahmad Banijamali*, Philipp Heisig‡, Johannes Kristan§, Pasi Kuvaja*, Markku Oivo*
*M3S research unit, ITEE Faculty, University of Oulu, Finland
‡IDiAL - University of Applied Sciences and Arts Dortmund, Germany
§Bosch Software Innovations GmbH, Germany

***Abstract*—Software architectures play an important role in the success of cloud platforms in automotive domain. To review the architecture designs in this context, it is necessary to explore different design decisions such as design styles, quality attributes (QAs), and evaluation methodologies.**

To this end, we aimed to investigate (i) architectural design styles, (ii) major QAs, and (iii) architecture evaluation methods that are applied in the cloud platforms in automotive domain. We conducted an online survey to collect data from participants in industry and academia. Methodologies, such as descriptive statistics and grounded theory, were used to analyse the data.

We collected 42 valid responses from participants with different roles, backgrounds, and years of experience. Considering the survey objectives, (i) event-driven and service-oriented architecture (SOA) were the most applied design styles to fulfil QAs. (ii) Availability, reliability, and security were the major QAs among other attributes. Finally, (iii) active reviews from intermediate design (ARID) and the scenario-based architecture analysis method (SAAM) were mostly applied when evaluating the architecture of cloud platforms in automotive domain.

The results of our survey show a spectrum of different applicable design styles, QAs, and evaluation methods. For selecting the set of architectural design decisions, one should consider the business scenarios and relevant quality requirements that should be supported by the cloud platforms in automotive domain.

***Index Terms*—Software architecture design, Cloud computing, Automotive, Internet of things (IoT)**

I. Introduction

Connecting vehicles to external objects, services, and platforms has enabled more business scenarios [1] for secure data exchange beyond vehicles [2], remote monitoring, and over-the-air software maintenance [3]. In achieving reliable solutions, the architectural design of connected vehicles benefits from advances in cloud platforms, internet of things (IoT), and communication technologies [4]. As for nearly real-time services, it is also necessary to bring data processing from the cloud to the edge of the network and closer to the vehicles and devices through distributed tiny clouds that is known as fog computing, which enables new services by the mass adoption of IoT [5].

Designing architectures of cloud platforms in automotive domain (ACPs) (Fig.1) includes new challenges compared to other IoT cloud platforms. For example, architectural decisions in ACPs should consider a large number of heterogeneous vehicles and devices, various communication technologies that are specific to different providers, longer life cycles of vehicles, less IT standardisation, less reliable connectivity, a bigger variety of connectivity hardware, and scalable real-time services in safety-critical systems [1], [6].

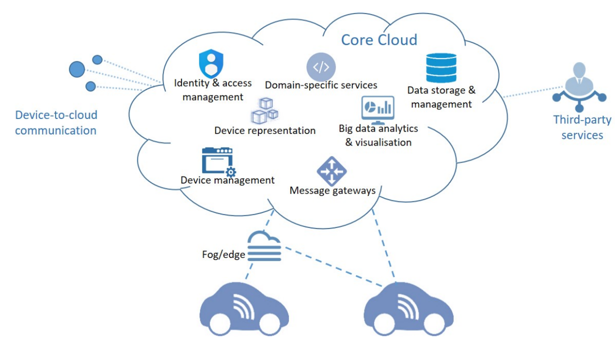

Fig. 1. Cloud platforms in automotive domain

In this area, software architects' design decisions is an important research topic [7] that makes a significant impact on system qualities [8]. Although previous studies [6], [7] emphasised the research gap on design decisions such as QAs and design styles in ACPs, there are scarce studies that investigate architectural design styles, QAs, and evaluation methodologies in the context of ACPs.

To this end, we conducted an online survey on software architecture design in ACPs to investigate (1) architectural design styles and selection criteria, (2) quality attributes (QAs) and their evaluation methods, and (3) architecture evaluation methods and relevant challenges. The scope of this study is on the *software architecture design of core cloud* (see Fig.1) and we excluded from our survey other software knowledge areas [9], in-vehicle platforms, and communication networks.

The findings of our survey are valuable for industrial practitioners to look for information on design styles in terms of QAs and architecture evaluation methods in ACPs. In addition, researchers could gain insight into the design criteria and challenges behind software architectures in ACPs.

This research was supported by the ITEA3-APPSTACLE research project and funded by Business Finland.

Corresponding author: Ahmad Banijamali (ahmad.banijamali@oulu.fi)

II. Background

The first part of this Section provides a brief overview of emerging trends for ACPs. Next, it explores a few relevant studies on the architecture design of ACPs.

A. Emerging trends for ACPs

Integration of technologies such as cloud computing, wireless and RFID sensor networks, and in-vehicle sensors [6] has enabled more innovative business scenarios [1] in the automotive domain. For instance, automotive software has traditionally been developed and maintained separately by each car manufacturer in-house [10]. However, this approach could not meet the long-term challenges of the industry, as the diverging strategies of car manufacturers could not provide solutions for the increasing number of connected devices [11]. Therefore, new international collaborations between large car manufacturers, tier-1 suppliers, and cloud platform and solution providers have been formed up (e.g., [12], [13]) to develop ACPs addressing mass-differentiation in automotive services.

Nevertheless, there has been an increased focus from other companies and research institutes on introducing new solutions for centralised vehicular data analysis and over-the-air software maintenance. For example, APPSTACLE[1] is a European Union (EU) project that seeks to create an open software platform to connect vehicles to the Internet and cloud platforms. The project outcome, which is publicly available via the Eclipse Kuksa project, interconnects a wide range of vehicles to an open cloud platform through open in-vehicle and Internet connection interfaces. CarCoDe[2] is another example of a software platform developed for the automotive domain to enable a new traffic-service domain ecosystem. It created the opportunity for third-party developers to generate more innovative automotive applications.

B. Software architecture of ACPs

Architecture design in ACPs is still in the early stages of research [7]. To address this area, prior studies have applied various architectural design styles, such as multi-layered [14], [15] and service-oriented architecture (SOA) [16], [17], to address quality concerns in ACPs. For example, Serrano et al. [18] developed a generic data-processing architecture for real-time traffic-based routing to address scalability and resiliency in this area. They argued that the proposed software architecture could serve a wide range of workloads and use cases with low-latency requirements. Another study [15] proposed a multi-layered context-aware architecture. The authors explained that vehicular social networks, context-awareness, and security are essential for opening up emerging services such as real-time traffic information prediction [15].

Fiosina et al. [19] proposed real-world scenarios of intelligent traffic system applications and demonstrated the need for next-generation big data analysis and optimisation strategies. The proposed architecture supports scalability, service encapsulation, dynamic configuration, and on-demand delivery of big data in the intelligent transportation area.

Due to the technological variety in ACPs, architecture designs must assure stakeholders [8] that provisional services will meet the quality requirements at a specific level of cost and risk [20].

[1]https://itea3.org/project/appstacle.html
[2]https://itea3.org/project/carcode.html

III. Research Method

We adopted survey research methodology for this study. Our survey collected qualitative and quantitative data to provide a "snapshot" [21] of the current status of software architecture design in ACPs. For this study, we applied the survey process of Ciolkowski et al. [22] and the activities proposed by Pfleeger and Kitchenham [23]. Fig.2 shows the survey process in this study that will be described in greater detail below.

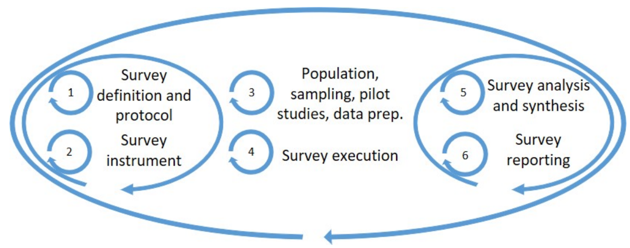

Fig. 2. Survey process

A. Objectives and research questions (survey definition)

Our research aims to investigate software architecture design in ACPs. Software architecture is a broad topic that addresses different topics such as design decisions, system representations, performance quality, re-usability, and business success [8], [24]. This study focuses on design styles as a collection of architectural design decisions [25], QAs as the system capabilities and behaviour to meet end user expectations [8], and architecture evaluation methods as a means of checking whether the decisions made were right [26]. The research questions (RQs) of this study are summarised in Table I.

TABLE I
Research questions and their rationales

Research question	Rationale
RQ1: What architectural design styles are used in ACPs to fulfil quality attributes?	To study major design styles and selection criteria to satisfy QAs.
RQ2: What are the important quality attributes in ACPs?	To explore major quality attributes and relevant evaluation methods.
RQ3: What architecture evaluation methods are applied in ACPs?	To explore the architecture evaluation methods and challenges.

B. Survey design

1) Form of the survey: To present characteristics of a population in ACPs, we adopted a descriptive approach [27]. It reviews which design styles, QAs, and architecture evaluation methods are applied [28] in the context of ACPs. For this purpose, we used an online questionnaire as a survey instrument as it allowed for data collection from globally diverse locations in a time- and cost-effective way [29].

2) Population, sampling method, sample size, and participant selection: The survey population included (1) industrial practitioners and (2) academic researchers with practical experience in software architecture design in the context of ACPs. The survey instrument included several questions to identify the roles, years of experience, and working areas of the participants.

Botta et al. [30] noted that convergence of cloud computing and the IoT is a recent and promising research trend introduced after 2010. In the same way, the design of ACPs is a more recent research topic, so there is a rare opportunity to find practitioners with more than 10 years of experience in all topics of this study (cloud computing, IoT, and automotive).

To address the concern about whether the survey responses can reflect the existing architecture knowledge on ACPs, we considered the following criteria regarding which respondents to include:

- To emphasise respondents with a real industrial background, we excluded all participants who did not have any practical experience either in general or in software architecture design.
- All included participants had work experience in at least two of the following three topics: cloud computing, IoT, and automotive. Software architecture engineering experience was considered mandatory.
- Respondents with a "researcher" role could come from industry and work in, for example, research and development departments (R&D).
- Collaboration between industry and academia in many national or international projects (e.g., ITEA3-APPSTACLE and ECSEL-AFarCloud) made researchers and practitioners to work closely together in real industrial use cases and projects. Thus, we included academic researchers who met the first two inclusion criteria.

We applied a non-probabilistic approach to extract a sample of the whole population [31]. It was not practically possible for us to conduct probabilistic sampling because there was no previous knowledge about the entire survey population. In addition, it was not feasible to conduct systematic sampling after the survey because the data in the survey were entirely anonymous. We did not collect company names or any revealing information, which could have damaged the quality of the data [32], as some of the participants may not have been allowed to legally disclose organisational data.

In this survey, we adopted the convenience sampling method [28], [32]. It dictates that *"subjects are selected because of their convenient accessibility to the researcher. This technique is easy, fast, and usually the least expensive. The criticism of this technique is that bias is introduced into the sample"* [33]. Due to challenges such as the large population of software engineering participants, it was impossible to access every individual in the sample. Therefore, most researchers rely on sampling techniques such as convenience sampling, which is a widely accepted methodology in software engineering research [28] and, in our case, was the best feasible choice.

We reached the target population using the following methods:

- The authors' contacts (i.e., existing or previous co-workers and co-authors);
- Automotive software engineering communities, groups, or projects; for this purpose, three separate approaches were adopted: (i) automotive software engineering communities (i.e., automotive-grade Linux [AGL]); (ii) IoT/cloud/automotive participants in social media (i.e., LinkedIn groups); and (iii) coordinators and technical members in relevant EU projects; and
- The academic researchers from prior literature reviews, conferences, or symposia.

We created a follow-up plan, which involved sending reminders to the target population to encourage more people to participate in the survey.

C. Data preparation and collection

1) Survey instrument: We constructed the survey instrument[3] with the help of previous literature and surveys on the software architecture engineering, IoT, and cloud platforms (e.g., [30], [34], [35], [36]). We did not want to frame the questions [37] by designing a domain-specific questionnaire, as our understanding of the domain might have influenced the answers and increased the research bias. Thus, we decided to keep the questions generic and let the respondents shape the architecture knowledge on ACPs, which was the main interest of paper.

The survey construction was required to adequately cover the scope of this study while at the same time optimise the number of questions. The instrument had 19 survey questions (SQs): 10 questions on demographics and 9 specific questions. We used the questions on demographics to understand the participants' backgrounds and their familiarity with the major topics in the survey. However, we did not include the demographic questions in RQs of the paper, yet it was necessary to have a good understanding of the participants' backgrounds in this emerging domain so that we could filter valid responses according to our inclusion criteria (see Section III-B2).

The questionnaire included a combination of closed questions (pre-designed answers) and open questions (free text). We used a five-point Likert scale for closed questions to score the responses. Multiple-choice questions allowed the respondents to vote differently for design styles, QAs, or architecture evaluation methods instead of selecting only one option. It enabled us to draw a spectrum of different applicable options.

2) Survey format: The survey opened with a welcome page, including aim of the survey, the estimated completion time of the survey, and the confidentiality of the responses. The survey was hosted on an online survey tool provided by Webropol.[4]

[3]https://www.dropbox.com/s/rczzjaq85l8eix2/survey.pdf?dl=0
[4]https://link.webropolsurveys.com/S/7BF39BC80144BFDD

D. Survey execution

The survey was open to the participants for three months, from April to June 2018. We asked the participants to complete the survey online. Participation was voluntary and anonymous. To maximise the outreach of the survey, we sent invitation emails to various software communities (e.g., IoT, cloud, and automotive communities). These software communities were contacted through mailing lists in which it was not possible to check the number of active recipients in each community. We applied snowball sampling [38], where the email recipients were also asked to forward the survey link to other people with relevant expertise for this survey. Thus, it was not possible for us to calculate the response rate (i.e., the ratio of the people who received the invitation to those who filled out the survey) [39], as there was no information about how many times the survey link was forwarded.

E. Data analysis

We mapped each SQ to a relevant RQ. SQs 11-13 were used to answer RQ1, SQs 14-16 for RQ2, and SQs 17-19 addressed RQ3. We did the data analysis using descriptive statistics for the closed questions. We applied grounded theory [40] to the open questions to create different categories out of the responses received and used the classification by Adolph et al. [41] to code the data input in the open questions. We used open coding to generate different concept clusters, while selective coding helped to identify the core categories to demonstrate the variation in the data. At least two authors were involved in the data analysis process.

F. Protocol review and pilot studies

The survey protocol[5] in this study was iteratively reviewed by all the authors. To evaluate the SQs [42], it was not efficient to use the entire population [39]. Therefore, we conducted three rounds of pilot studies on the draft sets of the SQs [42] to simulate the survey execution while reducing author bias. In total, two industrial practitioners and three experienced academic researchers participated in the pilot studies. During the pilot studies, we were able to improve the scope, questions, and terminologies used in this survey. In addition, we could estimate the time needed to complete the questionnaire. The participants in the pilot studies were representative of the potential respondents of this survey (see the inclusion criteria in Section III-B2). The first round of the pilot study was initiated with two experienced academics who were asked to fill out the survey draft. Next, we improved the questionnaire based on the feedback and suggestions from the first round. The same steps were followed with different participants in the second and third rounds until no further suggestions for improvement were offered.

IV. Survey Results

This section summarises and presents the findings collected from 42 valid responses.

[5]https://www.dropbox.com/s/un0an796911f6lm/protocol.pdf?dl=0

TABLE II
PARTICIPANTS' DEMOGRAPHIC CHARACTERISTICS

Educational level		**Academic background**	
Bachelor's degree	23.8%	Computer engineering	35.7%
Master's degree	57.2%	Software engineering	28.6%
Doctoral degree	19.0%	Information systems	11.9%
Organisational type		Communication & networks	11.9%
Large company	38.1%	Electrical engineering	7.1%
University & college	35.7%	Other	4.8%
SME	16.7%	**Organisational role**	
Research institute	9.5%	Software architect	28.6%
Country		Researcher (ind. and acad.)	28.6%
Germany	33.3%	Software developer	14.3%
India	19.0%	Project manager	9.5%
Finland	14.3%	Consultant and instructor	7.1%
Italy,France,Turkey,US (2p.)	19.0%	Product owner	4.8%
Chile,Colombia,Mexico, Netherlands,Russia,Spain(1p.)	14.3%	Senior manager (e.g., CTO)	4.8%
		Other	2.3%

A. Demographic data of the respondents

Table II presents the participants' background information, including their educational level, academic background, organisational type, and organisational role. Based on responses related to educational level, all participants had an academic degree and more than half of them had a master's degree. Computer and software engineering were two main areas of the participants' academic education. The survey respondents mostly worked at large companies, followed by universities and colleges, small and medium-sized enterprises (SMEs), and research institutes. The respondents were mostly software architects, researchers (coming either from industry or academia), and software developers. Germany had the most participants in the survey.

Fig.3 presents the participants' total work experience and their experience in the software architecture domain. The results show that more than half of the respondents (52.4%) had at least a decade of work experience. In addition, a large share of participants (66.7%) had more than 4 years of experience in software architecture. In a separate question, 66.7% of participants declared that they had received "professional training" related to software architecture engineering. We excluded all participants without architecture design experience from the survey.

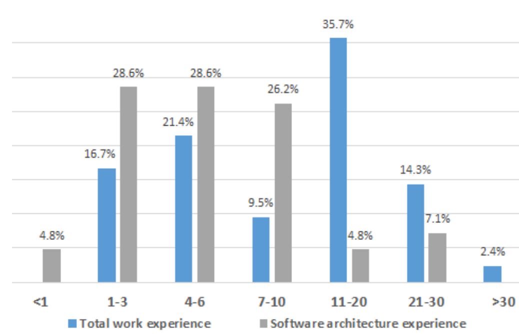

Fig. 3. Total and software architecture engineering work experience (in years)

Fig.4 shows the participants' experience in the IoT, cloud computing, and automotive domains. As we noted before, the participants were more specialised in one area than in the other two. For example, 68.4% of the participants with less than one year of automotive experience had more than one year of experience in the IoT, and 52.6% had more than one year of experience in cloud computing. The convergence of these areas is recent, and due to the rapid changes in ACPs, having respondents with a diverse range of experience was expected.

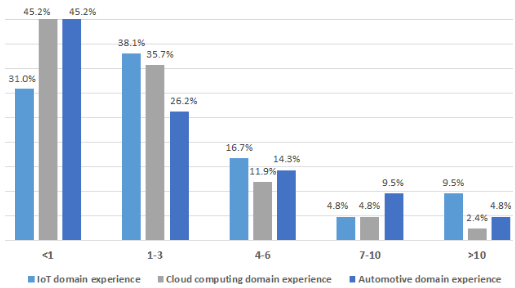

Fig. 4. Years of experience in IoT, cloud computing, and automotive areas

B. RQ1. What architectural design styles are used in ACPs to fulfil quality attributes?

This question attempts to present a wide range of architectural design styles relevant to QAs in ACPs. For this purpose, we collected a list of design styles from previous literature (e.g., [34], [43], [44]). A follow-up (free-text) question collected additional relevant styles from the participants.

According to a five-point Likert scale, event-driven was selected as the most applicable architectural style in terms of QAs in ACPs. The next applied design styles were service-oriented architecture (SOA), multi-layered architecture, and client-server architecture, followed by pipeline and microservice styles. In an open follow-up question, Lambda and message bus architecture were suggested by two respondents. Fig.5 presents a spectrum of architectural design styles, through which hybrid and slightly varied design styles are often expected to meet various scenario in the ACPs.

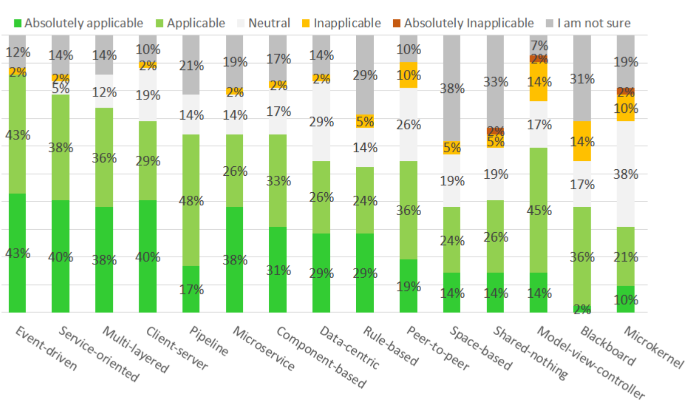

Fig. 5. The survey results on architectural design styles

1) Selection criteria for design styles: Fig.6 shows that more than half of the participants selected functional requirements (61.9%) and domain-specific QAs (59.5%) as the most important criteria when selecting design styles in ACPs. Although there are distinct QAs and functional requirements in ACPs, we could not distinguish different preferences for selection criteria from other IoT cloud platforms.

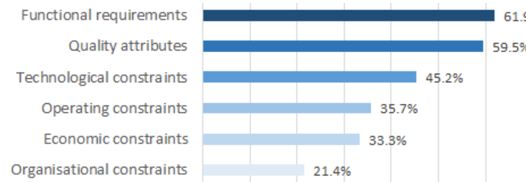

Fig. 6. The survey results on design style selection criteria

C. RQ2. What are the important quality attributes in ACPs?

Fig.7 shows a list of QAs that were collected from previous literature (e.g., [35]). A follow-up (free-text) question collected other relevant QAs. Although QAs are a broad topic for one question, the findings could be valuable for the selection of design styles and cloud technologies that improve QAs. Participants rated different QAs according to their importance (in a range from 1 as the least important to 10 as the most important) when designing the architecture for ACPs. We used these rates to calculate the weighted arithmetic mean ($\bar{x}$) for each QA. The responses where participants selected "I am not sure" were assigned a weight of zero. According to our calculation, availability ($\bar{x} = 6.7$) was the most important QA, followed by reliability ($\bar{x} = 6.5$), security ($\bar{x} = 6.3$), scalability ($\bar{x} = 6.0$), and interoperability ($\bar{x} = 6.0$). In the follow-up question, privacy, testability, compatibility, and flexibility were noted by participants as additional important QAs. Selecting important QAs does not necessarily mean that other QAs are redundant.

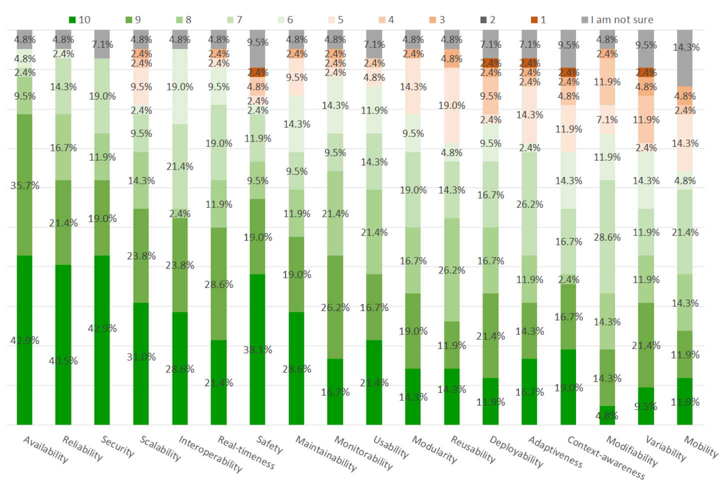

Fig. 7. The survey results on distribution frequency of QAs

1) Evaluation of ACPs with respect to QAs: We asked the participants which methodologies should be used to assess whether designed architectures meet the QAs in ACPs. A large share of respondents (42.9%) declared testing as a suitable approach, as it can be conducted at different levels.

Examples of testing include automated test-benches, stress tests (for performance-related attributes), hardware-in-the-loop (HIL) and software-in-the-loop (SIL) testing, controlled and live test suites, and user acceptance tests. Table III lists other evaluation methodologies noted by the participants.

TABLE III
EVALUATION METHODS IN ACPS WITH RESPECT TO QAS

Evaluation approach	Frequency
Testing and design validation	42.9%
User feedback monitoring	16.7%
Empirical studies (i.e., case studies and experiments)	11.9%
Simulation	9.5%
Evaluation according to functional requirements	7.1%
Statistical and mathematical (formal) methods	7.1%
Expert judgement	4.8%
Using software metrics approaches (e.g., the goal question metric [GQM])	2.4%

D. *RQ3. What architecture evaluation methods are applied in ACPs?*

We collected a few architecture evaluation methods from the literature (e.g., [36]) and asked the participants to add other relevant methodologies in an open follow-up question. Fig.8 shows that among various methods, active reviews from intermediate design (ARID) and the scenario-based architecture analysis method (SAAM) were applied the most in ACPs.

Additionally, we asked the participants about architecture evaluation challenges, as the availability and quality of architecture reviewers were reported by them (28.6 %) as major challenges in ACPs. A few participants (16.7%) reported that evaluation methodologies are not well-adapted with agile methodologies and that they are often time-consuming. They noted that it is challenging to verify that enough information to make architectural decisions has been collected, thus incurring technical debt left in the platforms (e.g., estimating performance in real scenarios with unknown technologies and complex analytics models). Finally, difficulties in evaluating and comparing other architecture design alternatives were reported by a few participants (7.1%).

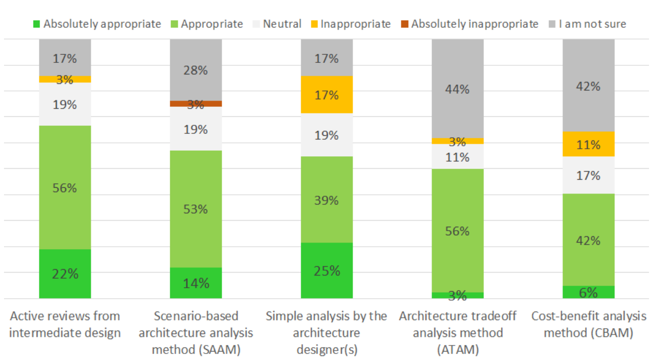

Fig. 8. The survey results on software architecture evaluation methods

V. DISCUSSION

1) Participants' demographic information: The results reflect a wide range of academic background and experience from participants. It shows that the software architecture of ACPs is an attractive topic for researchers from different educational or working backgrounds. We excluded participants without architecture design experience to create a viable snapshot of ACPs architecture knowledge. The good balance achieved between industrial practitioners (54.8%) and academic researchers (45.2%) meant that both scientific and practical knowledge were included in our findings.

2) Architectural design decisions vs. scenarios in ACPs: The results of this survey support the extensible architecture design decisions in ACPs to accommodate changes based on different scenarios. As shown in Section IV, we have a spectrum of different styles, QAs, and evaluation methods. This result indicates that we cannot select an optimum combination of design decision alternatives in ACPs and architects should apply different design styles with respect to QAs and functional requirements in ACP scenarios (see Fig.6). In addition, design decision alternatives dynamically change to reflect the new quality requirements that evolve in ACPs. A study [45] noted that we can rely on the feedback from the architecture evaluation to prioritise and create a new list of design decisions.

3) Inter-dependencies among design decisions: The survey results show inter-dependencies among different design decisions in ACPs. Loosely-coupled and distributed software components and services in ACPs can justify the selection of event-driven (as the first style selected) and client-server (as the fourth style selected). Event-driven architectures enable event processing between highly distributed services [46] in the cloud and vehicles. In addition, the survey results indicate that the first five major QAs selected in ACPs include availability, reliability, security, scalability, and interoperability. Previous studies [20], [47] emphasised the relation between these QAs and design styles such as SOA, multi-layered, and microservices. This justifies the participants' selection of SOA as the second, multi-layered as the third, and microservices as the sixth design styles to address major QAs in ACPs context.

ARID and the SAAM were the most applied architecture evaluation methods among the participants. The survey participants mentioned that ARID is used to review the viability of the design strategies. They also declared that the ACPs is a recent and emerging topic and when using the SAAM, often, only a few scenarios are described in detail; thus, concerns remain about whether the architecture is representative of all sort of expected or unexpected scenarios.

4) Emerging new QAs in ACPs: Previous research [30] discussed emerging QAs in ACPs compared to other IoT cloud platforms. For example, security concerns such as spoofing user identity, tampering, and repudiation are a recent topic and are addressed through architectures of ACPs to avoid security issues that lead to safety and privacy issues. The relation between security and design styles such as multi-layered, client-server, and microservices was noted before [48], [49]. Our survey results also emphasised the importance of security (as the third QA selected) in ACPs and its relevant design styles (the third, fourth, and sixth styles selected).

Scalability is a challenge in ACPs when handling a dynamically changing number of vehicles and devices caused by various traffic situations. Availability and reliability are other QAs highlighted in our survey results for the critical scenarios such as when vehicles leave the network coverage. Design styles often provide helps on how to solve most of the quality challenges but can be incomplete or ambiguous [50].

5) Future studies: The large number and variety of QAs and design styles in the context of ACPs is an interesting observation from our survey. Researchers could explore other aspects of architecture design in ACPs, such as architectural activities and notation languages. It would also be interesting to conduct an in-depth analysis of how to apply design styles in terms of QAs in ACPs. Real-time architecture evaluation methods for reliability and performance of the ACPs have to be further investigated.

A. Threats to validity

There are potential threats to the validity of this research that could bias the results [51]. This section elaborates on the strategies used to minimise the impacts of those threats according to the guidelines presented by Wohlin et al. [52].

1) Internal validity: There was no causal relationship between the variables and outcome of the survey, and the internal validity threats were not discussed.

2) Construct validity: Second category of threat concerns whether the survey constructs were designed correctly. In order to construct the survey instrument, the authors conducted multiple rounds of pilot studies to explore how different participants interpreted the questions and terms used in the survey. The feedback from the pilot studies were implemented to improve and adapt the survey according to the terms that were understandable in both industry and academia.

We provided several examples and guides (as web links) for the questions in the survey instrument. The intention was to help the respondents to better understand the questions and reduce the discrepancy between the authors' and respondents' mind-sets.

The survey protocol was reviewed by all authors and discussed in joint meetings. This reduced the conflicts of understanding between the authors and mitigated the risk of an unclear survey design and irrelevant questions.

Another concern about surveys is that the questions may not be truthfully answered by the respondents. There are reasons for this phenomenon, such as closed questions that reduce the participants' liberty in making responses. To mitigate this threat, each closed question was followed by either an open follow-up question or an "other" option where respondents could add extra inputs. We also made the responses anonymous and assured the respondents about the confidentiality of the survey by informing them about the use of a trusted survey hosting service. The survey clarified that individual results would not be published.

3) External validity: This survey adopted a convenience sampling method, which may have increased the risk that the sample did not reflect the whole population. Therefore, we did not limit the survey participants to our personal contacts. Instead, the authors shared the link in different IoT, cloud, and automotive software communities, at software engineering conferences, and with relevant EU projects. In addition, we defined criteria to only include valid responses into the survey results. This ensured that the final pool was randomly selected and comprised relevant backgrounds in the context of ACPs.

4) Reliability: A potential threat to the validity of the research results was related to the researchers' bias in the extraction and interpretation of the survey data. To minimise the effects of this threat, a data extraction form was created in Microsoft Excel to generate valid categories from the responses to the open questions. The form was reviewed and analysed by two authors.

The survey results are based on 42 valid responses that were selected based on our inclusion criteria. Our experience from similar surveys (i.e., [53], [54]) indicated that the current number of participants was sufficient for the reliability of the results and generalisability of the conclusion. Nevertheless, future studies could improve the credibility of the results with other complementing methodologies, such as expert interviews.

VI. Conclusion

ACPs have received increasing attention from research and industrial communities. To capture existing architectural design knowledge and experience in ACPs, we distributed an online survey on design styles, QAs, and architecture evaluation methods for ACPs. We collected 42 valid responses that were further analysed.

Our findings showed that the participants consider major QAs in the context of ACPs such as high availability, reliability, scalability, and interoperability to apply the relevant design styles (i.e., event-driven and SOA). ARID is applied before a comprehensive architecture evaluation, while the SAAM is to review the sets of QAs in the form of scenarios in ACPs.

The survey results are valuable for both ACP's researchers and practitioners as the investigation of design styles in terms of QAs is also concerned about specific design contexts.

References

[1] S. K. Datta, M. I. Khan, L. Codeca, B. Denis, J. Härri, and C. Bonnet, "Iot and microservices based testbed for connected car services," in *2018 IEEE 19th Int. Symp. on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*. IEEE, 2018, pp. 14–19.

[2] B. P. Rao, P. Saluia, N. Sharma, A. Mittal, and S. V. Sharma, "Cloud computing for internet of things & sensing based applications," in *Sensing Technology (ICST), 2012 6th Int. Conf. on*. IEEE, 2012, pp. 374–380.

[3] S. K. Datta, R. P. F. Da Costa, J. Härri, and C. Bonnet, "Integrating connected vehicles in internet of things ecosystems: Challenges and solutions," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016 IEEE 17th Int. Symp. on*. IEEE, 2016, pp. 1–6.

[4] S. K. Datta, J. Haerri, C. Bonnet, and R. F. Da Costa, "Vehicles as connected resources: Opportunities and challenges for the future," *IEEE Veh. Technology Mag.*, vol. 12, no. 2, pp. 26–35, 2017.

[5] M. S. de Brito, S. Hoque, T. Magedanz, R. Steinke, A. Willner, D. Nehls, O. Keils, and F. Schreiner, "A service orchestration architecture for fog-enabled infrastructures," in *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 2017, pp. 127–132.

[6] W. He, G. Yan, and L. Da Xu, "Developing vehicular data cloud services in the iot environment," *IEEE Trans. on Ind. Informatics*, vol. 10, no. 2, pp. 1587–1595, 2014.

[7] S. Frey, L. Charissis, and J. Nahm, "How software architects drive connected vehicles," *IEEE Software*, vol. 33, no. 6, pp. 41–47, 2016.

[8] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice, 3rd ed.* Pearson Education, 2013.

[9] P. Bourque, R. E. Fairley *et al.*, *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0.* IEEE Computer Society Press, 2014.

[10] A. Pretschner, M. Broy, I. H. Kruger, and T. Stauner, "Software engineering for automotive systems: A roadmap," in *2007 Future of Software Eng. FOSE '07*. Washington, DC, USA: IEEE Comput. Soc., 2007, pp. 55–71.

[11] J. Contreras, S. Zeadally, and J. A. Guerrero-Ibanez, "Internet of vehicles: Architecture, protocols, and security," *IEEE Internet of Things Journal*, 2017.

[12] "Bmw and ibm team up for cloud-connected cardata network," Available: https://www.ibm.com/blogs/cloud-computing/2017/06/16/bmw-ibm-cloud-cardata/.

[13] F. Svahn, L. Mathiassen, and R. Lindgren, "Embracing digital innovation in incumbent firms: How volvo cars managed competing concerns." *Mis Quarterly*, vol. 41, no. 1, 2017.

[14] J. Wang, J. Cho, S. Lee, and T. Ma, "Real time services for future cloud computing enabled vehicle networks," in *Wireless Comm. and Signal Processing (WCSP), 2011 Int. Conf. on*. IEEE, 2011, pp. 1–5.

[15] J. Wan, D. Zhang, S. Zhao, L. Yang, and J. Lloret, "Context-aware vehicular cyber-physical systems with cloud support: Architecture, challenges, and solutions," *IEEE Comm. Mag.*, vol. 52, no. 8, pp. 106–113, 2014.

[16] R. Mietzner, F. Leymann, and T. Unger, "Horizontal and vertical combination of multi-tenancy patterns in service-oriented applications," *Enterprise Inform. Syst.*, vol. 5, no. 1, pp. 59–77, 2011.

[17] M. Yang, M. Mahmood, X. Zhou, S. Shafaq, and L. Zahid, "Design and implementation of cloud platform for intelligent logistics in the trend of intellectualization," *China Comm.*, vol. 14, no. 10, pp. 180–191, 2017.

[18] D. Serrano, T. Baldassarre, and E. Stroulia, "Real-time traffic-based routing, based on open data and open-source software," in *Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on*. IEEE, 2016, pp. 661–665.

[19] J. Fiosina, M. Fiosins, and J. P. Müller, "Big data processing and mining for next generation intelligent transportation systems," *Jurnal Teknologi*, vol. 63, no. 3, pp. 23–38, 2013.

[20] L. O'Brien, P. Merson, and L. Bass, "Quality attributes for service-oriented architectures," in *Proc. of the Int. Workshop on Systems Development in SOA Environments*. IEEE Comput. Soc., 2007, p. 3.

[21] C. Wohlin, M. Höst, and K. Henningsson, "Empirical research methods in software engineering," in *Empirical Methods and Studies in Software eng.* Springer, 2003, pp. 7–23.

[22] M. Ciolkowski, O. Laitenberger, S. Vegas, and S. Biffl, "Practical experiences in the design and conduct of surveys in empirical software engineering," in *Empirical Methods and Studies in Software eng.* Springer, 2003, pp. 104–128.

[23] S. L. Pfleeger and B. A. Kitchenham, "Principles of survey research, part 1: Turning lemons into lemonade," *ACM SIGSOFT Software Eng. Notes*, vol. 26, no. 6, pp. 16–18, 2001.

[24] ISO/IEC:42010, "Systems and software engineering - recommended practice for architectural description of software-intensive systems," *ISO/IEC Standard*, 2011.

[25] N. Medvidovic and R. N. Taylor, "Software architecture: Foundations, theory, and practice," in *Software Eng., 2010 ACM/IEEE 32nd Int. Conf. on*, vol. 2. IEEE, 2010, pp. 471–472.

[26] N. Rozanski and E. Woods, *Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives*. Addison-Wesley, 2011.

[27] D. E. Gray, *Doing Research in the Real World*. Sage, 2013.

[28] B. A. Kitchenham and S. L. Pfleeger, "Personal opinion surveys," in *Guide to advanced empirical software eng.* Springer, 2008, pp. 63–92.

[29] T. C. Lethbridge, S. E. Sim, and J. Singer, "Studying software engineers: Data collection techniques for software field studies," *Empirical Software Eng.*, vol. 10, no. 3, pp. 311–341, 2005.

[30] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: A survey," *Future Generation Comput. Systems*, vol. 56, pp. 684–700, 2016.

[31] T. Punter, M. Ciolkowski, B. Freimut, and I. John, "Conducting on-line surveys in software engineering," in *Proc. of the 2003 Int. Symp. on Empirical Software Eng. (ISESE 2003)*. IEEE, 2003, pp. 80–88.

[32] R. M. Groves, F. J. Fowler Jr, M. P. Couper, J. M. Lepkowski, E. Singer, and R. Tourangeau, *Survey methodology*. John Wiley & Sons, 2011, vol. 561.

[33] T. R. Lunsford and B. R. Lunsford, "The research sample, part i: Sampling," *JPO*, vol. 7, no. 3, p. 17A, 1995.

[34] E. Cavalcante, J. Pereira, M. P. Alves, P. Maia, R. Moura, T. Batista, F. C. Delicato, and P. F. Pires, "On the interplay of internet of things and cloud computing: A systematic mapping study," *Comput. Comm.*, vol. 89, pp. 17–33, 2016.

[35] ISO/IEC:25010, "Systems and software engineering-systems and software quality requirements and evaluation (square)-system and software quality models," *ISO/IEC Standard*, 2011.

[36] L. Dobrica and E. Niemela, "A survey on software architecture analysis methods," *IEEE Trans. on software Eng.*, vol. 28, no. 7, pp. 638–653, 2002.

[37] B. C. Choi and A. W. Pak, "Peer reviewed: a catalog of biases in questionnaires," *Preventing chronic disease*, vol. 2, no. 1, 2005.

[38] L. A. Goodman, "Snowball sampling," *The annals of mathematical statistics*, pp. 148–170, 1961.

[39] B. A. Kitchenham and S. L. Pfleeger, "Principles of survey research, part 2: Designing a survey," *Software Eng. Notes*, vol. 27, no. 1, pp. 18–20, 2002.

[40] K.-J. Stol, P. Ralph, and B. Fitzgerald, "Grounded theory in software engineering research: a critical review and guidelines," in *2016 IEEE/ACM 38th Int. Conf. on Software Eng. (ICSE)*. IEEE, 2016, pp. 120–131.

[41] S. Adolph, W. Hall, and P. Kruchten, "Using grounded theory to study the experience of software development," *Empirical Software Eng.*, vol. 16, no. 4, pp. 487–513, 2011.

[42] B. A. Kitchenham and S. L. Pfleeger, "Principles of survey research, part 3: Constructing a survey instrument," *ACM SIGSOFT Software Eng. Notes*, vol. 27, no. 2, pp. 20–24, 2002.

[43] E. Croes and J.-H. Hoepman, "Software architectural styles in the internet of things," *Radboud University Nijmegen*, 2015.

[44] E. Majidi, M. Alemi, and H. Rashidi, "Software architecture: A survey and classification," in *2010 2nd Int. Conf. on Comm. Software and Networks*. IEEE, 2010, pp. 454–460.

[45] A. Koziolek, "Research preview: Prioritizing quality requirements based on software architecture evaluation feedback," in *Int. Working Conf. on Requirements Eng.: Foundation for Software Quality*. Springer, 2012, pp. 52–58.

[46] B. M. Michelson, "Event-driven architecture overview," *Patricia Seybold Group*, vol. 2, no. 12, pp. 10–1571, 2006.

[47] L. Bass, I. Weber, and L. Zhu, *DevOps: A software architect's perspective*. Addison-Wesley Professional, 2015.

[48] T. Bi, P. Liang, and A. Tang, "Architecture patterns, quality attributes, and design contexts: How developers design with them?" in *Proceedings of the 25th Asia-Pacific Software Engineering Conference (APSEC), Nara, Japan*, 2018, pp. 49–58.

[49] Y. Sun, S. Nanda, and T. Jaeger, "Security-as-a-service for microservices-based cloud applications," in *2015 IEEE 7th Int. Conf. on Cloud Computing Technology and Sci. (CloudCom)*. IEEE, 2015, pp. 50–57.

[50] T. Häberle, L. Charissis, C. Fehling, J. Nahm, and F. Leymann, "The connected car in the cloud: a platform for prototyping telematics services," *IEEE Software*, vol. 32, no. 6, pp. 11–17, 2015.

[51] B. A. Kitchenham and S. L. Pfleeger, "Principles of survey research, part 6: Data analysis," *ACM SIGSOFT Software Eng. Notes*, vol. 28, no. 2, pp. 24–27, 2003.

[52] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer Science & Business Media, 2012.

[53] D. Ameller, M. Galster, P. Avgeriou, and X. Franch, "A survey on quality attributes in service-based systems," *Software quality journal*, vol. 24, no. 2, pp. 271–299, 2016.

[54] Z. Durdik and R. Reussner, "On the appropriate rationale for using design patterns and pattern documentation," in *Proc. of the 9th Int. ACM Sigsoft conf. on Quality of software architectures*. ACM, 2013, pp. 107–116.