Knowledge Integration In Self-Organizing Teams

A Practice-Oriented Perspective

Bachelor of Science Thesis in Software Engineering and Management

Petroula Theodoridou

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Knowledge Integration In Self-Organizing Teams
A Practice-Oriented Perspective

PETROULA THEODORIDOU

Supervisor: Håkan Burden
Examiner: Francisco Gomes

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Knowledge Integration In Self-Organizing Teams:
A Practice-Oriented Perspective

Petroula Theodoridou
Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
petroulatheodoridou@gmail.com

***Abstract*—Self-organizing teams following agile ways of working are a common occurrence in the software engineering field. Adopting such practices is considered essential for practicing agile development as a whole [1][2]. This research, conducted in close cooperation with a team of similar setup, is a proof of concept for a proposed architectural solution and a step towards the validation of the framework utilized in order to produce said solution. Moreover, the study aims at enhancing this framework by adding new elements in the process of designing and developing software, that apply not only to experienced software engineers but mainly to newcomers entering the field.**

I. Introduction

Agile practices are nowadays becoming a common choice in the industry for the purpose of software development [1][3][4]. One could claim that a big challenge for new approaches and methodologies is their potential for being generalized solutions applied to different teams with consistent results. In order to test these methods from a validity perspective, there needs to be a testing environment to facilitate the process, as well as an objective.

Practicing agile can prove especially challenging when a project is distributed in self-organizing teams [5] across multiple organizations with different areas and levels of expertise [6]. There is no definite answer as to how the presence of the aforementioned differences can affect the outcome of a project. Tackling the diversity of knowledge within a team has been a well-known issue in previous research, with most solutions being time-consuming and focusing on an extensive dialogue among the team members [6].

A. *Definitions of terms*

Definitions of terms used in this paper are provided in this section.

- Artifact: the proof of concept for this research
- Implementation: the process followed in order to produce the artifact
- Framework: Extended SARAD[1] approach [7]
- Knowledge Base: The proposed architectural solution based on the framework

B. *Research Goals*

This research consists an investigation regarding the feasibility of a proposed architectural solution stemming from a framework introduced by Sundgren [8]. Moreover, based on the results of the aforementioned investigation, suggestions on the process are provided in order to facilitate a potential optimization of the framework.

In his paper, ”A structured approach to review and refine architecture documentation”, the author introduces the TrustMe scenario [8], a visualization of threats displayed on the screen of a vehicle’s infotainment system. The aforementioned scenario is part of the Second Road Fas 2 (SRF2) [9] research project. By utilizing the action research method while incorporating elements from the design science approach, the study results in a refined architectural solution for the overall SRF2 project and the scenario [8], including suggestions regarding its implementation. The author uses the SARAD approach in order to evaluate the project’s architectural documentation against the scenario description. The results of this evaluation are used in order to form an extension of SARAD that is considered more fitting for the overall structure of the SRF2 project. The study focuses on the process and the way of working from the perspective of a junior architect [8].

The proposed architectural solution regarding the TrustMe scenario is realized in the current research with the implementation of a mobile application, which in this case consists the artifact. Essentially, the produced artifact serves to point out the differences in relation to its original design. Based on those differences, the research aims at enhancing the extended SARAD approach proposed by Sundgren.

For the purpose of accomplishing the aforementioned goals, two research questions were formulated:

- RQ1: Which are the reasons for the differences between the artifact and the proposed architectural solution?
- RQ2: How can the framework be improved given the reasons found in RQ1?

The identified research questions address an existing problem within the software engineering (SE) domain: the gap between initial designs and implemented solutions. There exist prior efforts focusing on closing that gap by emphasizing on the alignment between architectural elements and their emerging source code and structure [10][11]. This implies that

[1] Structured Approach for Reviewing Architectural Documentation

said efforts take the feasibility of a project for granted and aim at accomplishing a level of consistency. This research differs mainly due to the fact that it approaches the gap between an initial design and its implementation from the perspective of organizational communication.

C. *Expected Contributions*

The contributions of the current research can be divided into two interconnected parts, namely technical and process-related.

1) Technical: From a technical point of view, the research involves the implementation of the TrustMe application. This process incorporates a variety of development tools and frameworks utilized in order to develop said application. The potential differences between the artifact and the proposed architectural solution introduced earlier in the section, are considered part of the technical aspect.

2) Process: The technical part of the research will be used to evaluate the process leading to the initial design of the application. Taking into consideration the current work as well as previous related research, this paper will discuss potential improvement points of the extended SARAD approach.

D. *Structure of the current document*

The study is mainly presented in an imperative way. That being said, it begins by describing the context of this research, the organizations involved and the theoretical background. This is followed by a step-by-step guide about the process of designing and implementing the artifact. The results of the process are utilized in order to extract information about what could potentially be improved in the suggested framework, for the purpose of closing the gap between an initial design and its produced artifact. The above is organized in different sections, starting with section II, which provides the background of this research including previous work on the subject. Section III refers to the research methodology followed in order to implement the artifact. In section IV, the study presents the results in the form of tools and libraries utilized along with the relevant design choices concerning the system. The outcome of this research is discussed in section V, which results in answering RQ1. This, in turn, consists the material for answering RQ2, by providing suggestions for improving the initial framework. The study concludes with a discussion on how it contributes to the software engineering domain as well as its potential for being a contributing element to future work.

II. Background

This section describes the context of this study as well as the relevance to its scientific domain. The starting point refers to the organizational context, namely the Second Road Fas 2 project and is followed by the setup of the simulation lab environment, where most of the tests for this research were performed. Additionally, this section covers previous related research on the subject, while in the end it provides an introduction to the TrustMe scenario.

A. *Second Road Fas 2 project*

The basis of the current research stems from SRF2: a joint research project scheduled to run from January 2014 until May 2017. The purpose of the aforementioned project is to establish new and quicker ways of development within the automotive sector. The use of simulation technology is expected to assist in shortening the overall duration of the development process [8]. Besides the technical aspect of the project, this initiative is thought to be a step closer to the creation of an open environment, where individuals outside of the automotive industry will potentially be able to contribute to its advancement [9]. The participants of SRF2 are the Swedish National Road and Transport Research Institute (VTI), Volvo Cars Corporation (VCC), Viktoria Swedish Information and Communication Technology (Viktoria ICT) and SemCon. The TrustMe application that this research is focusing on, is part of the overall SRF2 project.

The aforementioned participants consist self-organizing disjoint teams. That being said, one can describe their way of working as agile, with team meetings held every few weeks. The meetings would usually involve a discussion about the current status of the overall project, planning of next steps and potential problems affecting the work progress. The different participants of the SRF2 project have different levels and areas of expertise. Even though the team shares knowledge and engages into discussions that may overlap said expertise, each participant is responsible for different tasks.

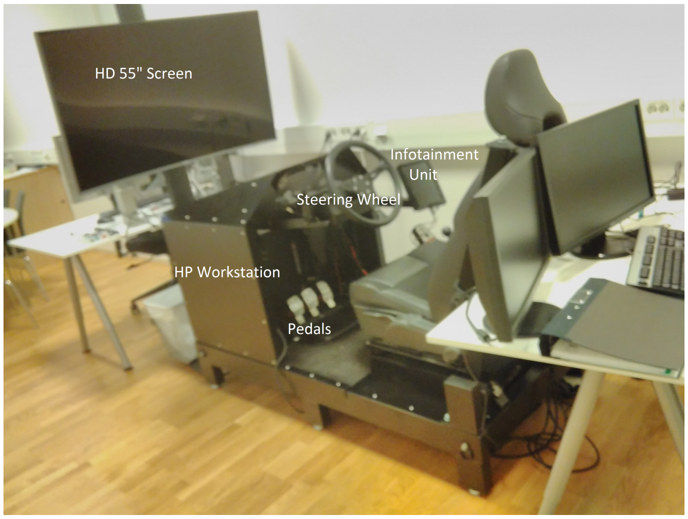

Fig. 1. HD 55” Screen, HP Workstation enclosed in the black frame, Steering Wheel, Pedals and Infotainment unit

B. *Open Innovation Lab*

The Open Innovation Lab, hereinafter referred to as OIL, is an environment equipped with a real-time simulator as well as multiple different modules that facilitate development and testing for the automotive sector. OIL’s technological platform was used in order to bring the implementation part of this research as close to a real-life scenario as possible.

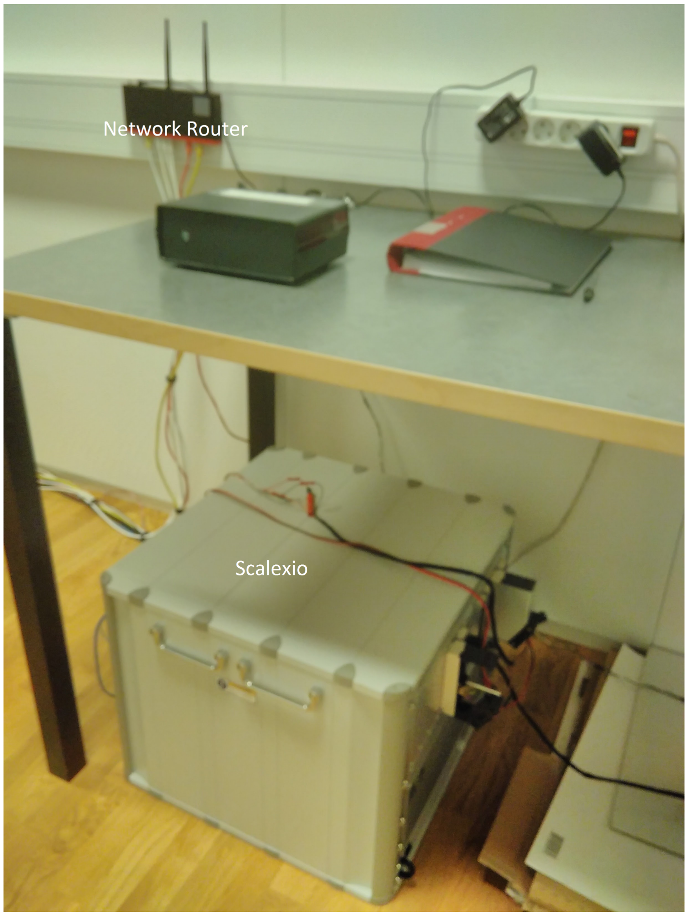

Fig. 2. Scalexio and Network Router

The modules present in the lab consist an integral part of the SRF2 project and consequently the realization of the TrustMe scenario. As discussed by Sundgren [8], said modules are deployed physically in the following way:

1) Environment Simulator: Deployed on an HP workstation. The visualization of the environment is displayed on a 55" screen.

2) Vehicle Simulator: Consists of a steering wheel, pedals, gear and modules simulating the XC90 vehicle model by Volvo. The core element of the vehicle simulator is the Scalexio, a hardware-in-the-loop simulator by dSpace [12].

3) Secure Gateway: Deployed on a Beaglebone black, the secure gateway module is handling the communication between the safety and non-safety critical parts of the vehicle.

4) Infotainment Unit: A 9" touch screen display paired with an Odroid C1+. The software running on the device is developed with AGA, an open source software platform providing the tools for developing in-vehicle applications [13]. Figures 1 and 2 illustrate OIL's setup.

C. Previous Research

This section provides the theoretical background of this research, by discussing previous work and approaches that relate to this study. It is divided into two sections: the first section reflects on the existing gap between architectural solutions and implementations while the second section provides a background of the SARAD approach and its extension.

1) Architectural solutions and Implementations: In an effort to address the issue of inconsistency between architectural elements and their respective implementations, Woods and Rozanski [10] suggest the creation of appropriate technologies adhering to a set of specific requirements. The authors justify the need for such tools by pointing out the weaknesses of existing ones, such as software architecture evaluation techniques, code generation and architectural description languages. Even though the aforementioned approaches relate to the identified issue, they are not thought to be providing solutions regarding the gap between an architecture and its implementation [10]. Moreover, the main problem with the aforementioned work on the subject is the fact that it focuses on aligning the architectures and their implemented solutions from a technical point of view.

Originally introduced by North [14], Behaviour-Driven Development (BDD) was a result of recurring issues arising from using Test-Driven Development (TDD) [15]. The use of natural language within the testing process is an addition of BDD which implies that the approach is not purely technical. Solis and Wang identify BDD as a blend of TDD, automated acceptance testing and ubiquitous language [15]. After conducting a literature review, the authors found that BDD toolkits emphasized more on the implementation stage of a project, without any support for the initial planning stage [15]. Essentially, this allows for an assertion that closing a potential gap between an initial architectural solution and its implementation cannot be accomplished with a standalone use of BDD techniques.

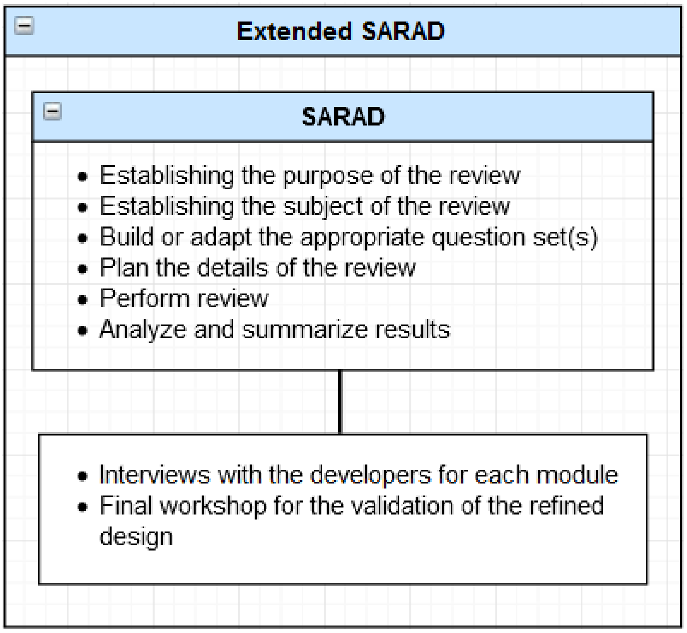
Extended SARAD
SARAD
• Establishing the purpose of the review
• Establishing the subject of the review
• Build or adapt the appropriate question set(s)
• Plan the details of the review
• Perform review
• Analyze and summarize results
• Interviews with the developers for each module
• Final workshop for the validation of the refined design

Fig. 3. The SARAD approach and its extension

2) SARAD approach and extension: SARAD is a six-step approach aiming at reviewing architectural documentation (AD). It is not considered a method yet [7] mainly due to the fact that it is not widely tested, but is considered by the authors a valid first step towards that direction. Emphasis is given on the difference between evaluating AD versus evaluating the architecture of a system. According to Nord et al. the latter is not the focus of the approach, since reviewing an architecture is a subject well covered by previous research. The six steps defining the SARAD approach are listed below [7]:

- Establish the purpose of the review
- Establish the subject of the review
- Build or adapt the appropriate question set(s)
- Plan the details of the review
- Perform review
- Analyze and summarize results

The aforementioned steps were extended by the addition of the following elements [8]:

- Interviews with the developers for each module
- Final workshop for the validation of the refined design

By conducting interviews with the different stakeholders as well as a final workshop, Sundgren proposed an extension of the SARAD approach. In essence, by applying SARAD he identified missing qualities in the system's specification and proceeded on a refinement of its architecture based on those qualities. This lead to an alignment of the AD and the DS[2] documents [8].

The starting point of the current research was to follow the author's proposed architectural solution for the TrustMe concept all the way to its implementation. This process resulted in a produced artifact but also in an evaluation of the way of working in self-organized teams. Consequently, the study provides insights and suggestions on how to improve the extended SARAD approach proposed by the author and thus assist in bridging the gap discussed in the previous section.

D. TrustMe

TrustMe is an AGA/android application that displays a vehicle's point of view. Essentially, it represents what a front facing camera would capture in real-time. That being said, it can be broken down into the following modules:

1) Video Streaming: The video streaming of this application utilizes the GStreamer library [16], an open source solution supporting the RTP protocol. The video stream is in Mjpeg format and is sent over UDP.

2) Threat identification: The values of the threat elements, namely the distance and the angle relative to the driver, are acquired by using MQTT, a publish-subscribe messaging transport protocol [17].

3) World to screen coordinates: The values received with the use of the MQTT protocol, translate to world coordinates and need to be transformed into screen coordinates in order for the threat(s) to be accurately displayed on the screen of the device.

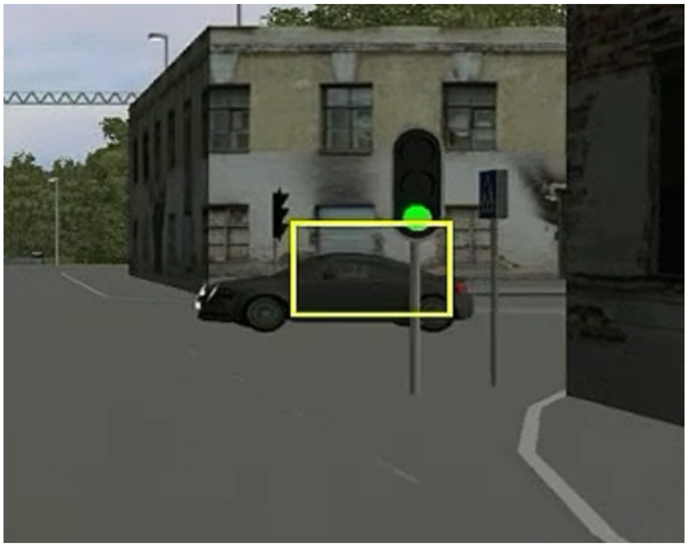

Fig. 4. TrustMe application proof of concept

4) User Interface (UI) and Overlay: The video streaming and the threat objects are independent of one another, as they result from different sources and there is no image processing in place. Both are displayed within the same window bounds, with the layer for the threat objects being on top of the video streaming layer.

The aforementioned breakdown structure is further analyzed in the following section.

III. RESEARCH METHODOLOGY

This section describes the methodology of this research step-by-step, as well as discusses the implementation of the TrustMe application. The research is based on the Design Science Research Methodology (DSRM), as introduced by Peffers et al. [18], with the addition of an extra element: the type of the study, discussed by Uysal [19]. The research is driven by the following research questions:

- RQ1: Which are the reasons for the differences between the artifact and the proposed architectural solution?
- RQ2: How can the framework be improved given the reasons found in RQ1?

DSRM has often been presented with the notion of not producing generalized knowledge but mere design artifacts that solve problems [20]. This study aims at placing the DSRM methodology by Peffers et al. in a practice-oriented context. By abiding to concrete definitions, the study follows the process of producing an artifact, in this case the TrustMe application, as well as targets the connection of said artifact to the organizational context of the SRF2 project. Thus, the goal becomes two-fold: to describe the process of implementing the TrustMe scenario and consequently discuss how the produced artifact is linked to the way of working as a member of a self-organizing agile team. The answer to RQ1 will stem from the result of implementing the TrustMe application. RQ2 will be answered based on the findings from RQ1 and will be analyzed in a different section.

[2] Demo-case Specification

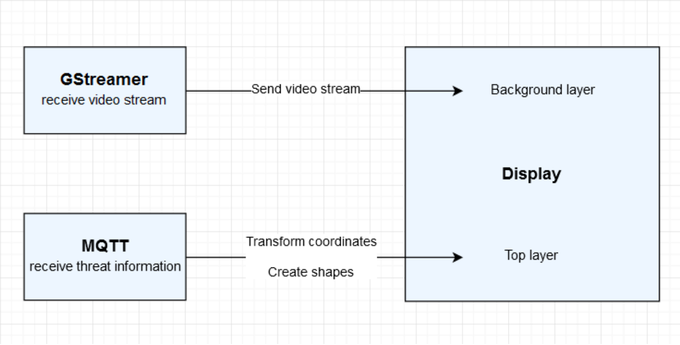

Fig. 5. Logical view of the application

A. DSRM

DSRM is a common framework introduced with the purpose of applying the design science approach to information systems. The current research utilizes the traditional DSRM framework and adds an extra step of recognizing the type of research. As discussed by Uysal, it is important for the research to be identified as theory-oriented or practice-oriented, as this distinction will affect its process. A practice-oriented research differs from a theory-oriented one in the sense that the former will result in an expansion regarding the knowledge of a specific organization [19] or individual [21]. What is more, the outcome of the study is not only expected to contribute to its relevant scientific domain but also to apply to the stakeholders' specific problem yielding clear improvements. On the other hand, a theory-oriented SE research would not potentially benefit an organization directly [19] but rather contribute with new propositions to the theory itself [21].

Based on the above context, the TrustMe application was implemented according to the following steps:

1) Type of Research: From the very first stages of this work, it became apparent that there needed to be a clear identification regarding the type of research to be conducted. The primary reason behind it, was the fact that the overall SRF2 project was focused mainly on implementation details and hence it was necessary for this research to be focused on the implementation of the TrustMe scenario as well. However, the goal of this study was not only to produce an artifact that would serve the needs of the organizations involved, but also to identify the process of how to apply a practice-oriented design science research methodology **(PDSRM)** in order to evaluate a proposed architecture.

2) Problem identification and motivation: As mentioned in a previous section, the problem and consequently the challenge of this research was to bring together two parts often separated from one another: the knowledge base and the artifact [20].

3) Defining the objectives of a solution: This step was necessary in order to create a structure for solving the identified problem. This is due to the differences that characterize the problem of the study and the objectives of a solution. In this research the problem was generalized, yet its solution specific. As discussed by Sundgren, the solution needed to adhere to specific software design constraints, the description of which was based on the author's research and will be briefly introduced here.

- **GStreamer**: An open-source library offering flexible and extended manipulation of streaming media [8], while offering the option of creating new plug-ins and filters [16]. It supports the Mjpeg format, the display of video stream with a screen grabber and allows for the use of the RTP protocol.
- **MQTT**: A messaging transport protocol that follows the publish-subscribe pattern. Essentially, a publisher broadcasts a message with a specific topic, to which, one or more subscribers can subscribe to. The publisher is not aware and does not need to know about the subscribers [22].

Figure 5 presents a high-level logical view of the application in relation to the aforementioned elements.

Besides the use of the above software, additional restrictions were imposed on the implementation of this project. Due to the fact that a previous attempt on using GStreamer on android resulted in a delay of approximately two seconds for the video streaming, it was suggested that this work involved native android, namely the use of the **Android NDK**. This translates to the use of a native code language (C/C++) for part of the application [23].

4) Design and development: This step involves the design of the artifact, its intended functionality and its development. In order for the solution to be easily understandable, the code was initially divided in packages representing different responsibilities. These packages served to the separation of core application elements, user interface elements, graph transformation functions and GStreamer libraries. The android native code resided in a different package as well, by default. The size of the initial application as far as classes used, can be considered small (<10). However, due to the code's complexity and the nature of the project it was deemed necessary to break it down from early on, based on content.

As mentioned in a previous section, the work was broken down in different parts, namely the Video Streaming, the Threat Identification, the World to Screen coordinates transformation and the UI Overlay. Process-wise and time-wise, this structure followed the overall SRF2 project. The team had a number of meetings every few weeks and discussed a specific part of this work, as well as work developed by the other team members under the SRF2 project. Since this process involved the communication with different stakeholders, it was necessary to, at times follow the overall schedule of the SRF2 project without the option of potentially speeding up the implementation of the TrustMe application. The different parts of said implementation, the division of which follows the aforementioned schedule, are extensively discussed below:

- **Video Streaming**: Following the initial requirement to use GStreamer in order to handle the streaming part of the application, as well as native android due to previously identified delay issues, there was a suggestion

TABLE I
VIDEO STREAMING

Send stream	gst-launch-1.0 -v -ximagesrc ! videoconvert ! clockoverlay ! videoscale ! video/x-raw, format=I420, width=800, height=600, framerate=20/1 ! jpegenc ! rtpjpegpay ! udpsink host=ipAddress[3] port=25000
Receive stream	udpsrc port=25000 ! application/x-rtp, encoding-name=JPEG, payload=26 ! rtpjpegdepay ! jpegdec ! autovideosink

to skip the use of the android framework in favour of a different one. Due to the fact that the attempt would be experimental regardless of the choice of framework, the research proceeded with what was already considered a challenging task: utilizing the android framework in order to setup the video streaming part of the application.

Instead of starting from scratch, the idea was to reuse one of the existing android examples from the official GStreamer documentation [24]. Then by modifying that sample, one could possibly adjust it to the specific needs of the application. Custom modifications as well as further additions had to be made in both the C part of the code and the Make files [23] in order to achieve an acceptable solution for this part of the project. An acceptable solution according to the initial specification would be a video streaming application without any significant human perceivable lag.

Even though the testing of the video streaming functionality could be done outside of OIL by utilizing a screen grabber and stream a computer's GUI on the mobile device, the application still needed to be tried out with the simulation setup, in order to identify and eliminate any potential performance issues. The initial result showed that there was a network configuration external to the TrustMe application that slowed down the video streaming. This issue was resolved shortly afterwards by SemCon.

The commands used in order to send the video stream from the computer to the mobile device, are illustrated in table I. While the statement used to send the stream is run on the simulation computer, the receiving end resides within the application, as part of building the required pipeline [16].

- **World to Screen coordinates transformation**: Due to the fact that the simulation environment represents the real world, in order to accurately display coordinates from the real world on a mobile device, a set of calculations and matrix transformations has to be considered. The details of this step are considered out of scope for the purpose of this research. However, it is worth mentioning that this process revealed errors in the matrix transformation setup that led to a small offset for the position of the shapes on the screen. This was not considered critical and did not impact the demonstration of the application, but since it was resolved by the respective organizations after the demonstration took place, it can be considered as a result that deviates from the proposed architectural solution and thus will be discussed further in the results section. The steps followed in order to manage the world-to-screen coordinates transformation included the following:
  - Obtaining the world coordinates
  - Translation by a given vector
  - Rotation by a given angle
  - Extending the rotation vector with a fourth element
  - Multiplication with a projection matrix
  - Normalization of the resulting x and y values
  - Calculation of screen coordinates
- **Threat Identification**: As threat identification here is considered the process of utilizing the MQTT framework in order for the application to receive information regarding the current threats. The MQTT functionality was implemented inside an asynchronous task, which provides an efficient way of handling background operations and updating the UI. Similarly to a previous step, an initial implementation was tested with a computer and a mobile device outside of OIL and it involved sending a message from the computer to the phone, in order to make sure the basic functionality exists. Further tests in the lab ensured that the application was able to constantly receive values for threats. Work on this step as well as the previous one was done during the same period of time, mainly due to the need for synchronization between the two. Initially the application was able to successfully receive the values of the threat objects, however signals of cars that were not visible on the screen would also be sent to the application. This issue was further investigated and solved by the respective organization (Viktoria ICT).
- **UI Overlay**: Displaying both the video stream and the threats on the screen required two layers: the drawing of threats was performed on top of the video streaming layer.

In order to test the overall functionality of the application, static background images and fixed values were used at first and the car in motion afterwards.

5) Demonstration: The demonstration of the artifact involves all meetings that took place throughout the course of this project, including the final demonstration that focused on testing the overall functionality of the TrustMe application, as well as work done by other members and organizations within the SRF2 project.

This step can be thought as being interlinked with the previous steps of the process. This is mainly due to the

[3] IP address of the mobile device

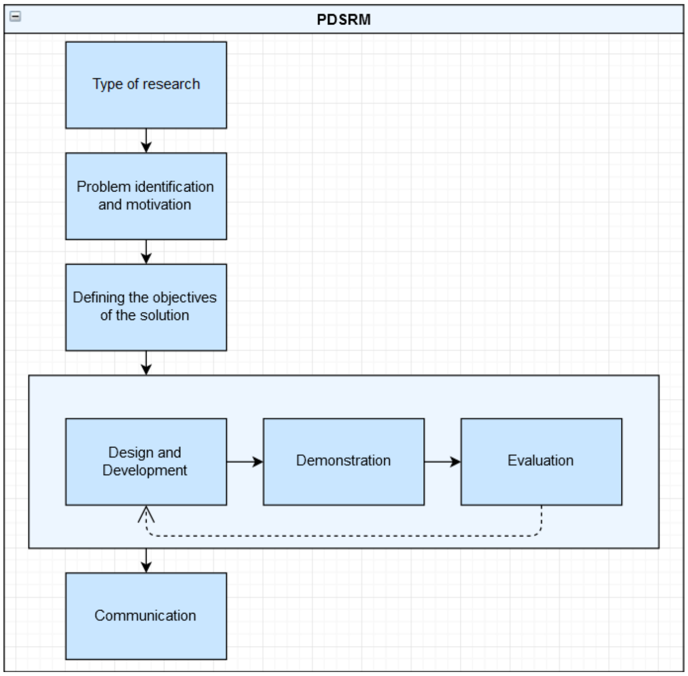

Fig. 6. TrustMe implementation process

fact that the implementation part of this research was tightly linked to the process of the overall SRF2 project. In other words, each part of the application had its own iteration cycle of Design/Development, Demonstration and Evaluation, as depicted in figure 6.

6) Evaluation: The produced artifact was evaluated in two ways discussed by Peffers et al. [18]: against the objectives of the solution and the feedback from the involved organizations. The application fulfilled the requirements set in the objectives section for being a proof of concept artifact and successfully met the performance standards set by the stakeholders.

7) Communication: Part of the topic was explored in *A structured approach to review and refine architecture documentation* [8], where the author discusses the SARAD approach and introduces the TrustMe scenario. The current study aims at presenting an overall view of the topic and its connection to previous research.

IV. Results

The design choices and the tools used for the implementation of the TrustMe scenario are the focus of this section. The first two subsections discuss the design of the application and the APIs that were utilized throughout the process. The third subsection focuses on the produced artifact and introduces a connection to the discussion part of this research.

A. Design

This section refers to the design of the application and briefly provides a motivation behind some of the design choices. The core aspects of the system are divided into different categories as proposed by Bass et al. [22], depending on the requirements they fulfill. This implies a division between functional requirements, quality attributes and constraints.

1) Functional Requirements: The following requirements are thought to be essential for the system to be considered functional:

- Display video stream: The application should provide a real-time representation of the vehicle simulator's surrounding environment.
- Display threats: The video stream should be combined with colored boxes displayed on top of identified vehicles (threats).

2) Quality Attributes: Since this application was not tailored for commercial purposes or further adjustments but was intended to be a proof of concept, most of the focus regarding the desired qualities of the system was on performance, reliability and simplicity. It was deemed necessary for the application to be responsive and fast. This was due to the fact that its main purpose was to represent a real-time scenario as well as be demonstrated in front of multiple different companies and interested parties. The experimental nature of this project was a strong indicator that some parts of the system should also be simplified. Since the SRF2 project consists of various organizations specialized in different areas, it was clear from the very beginning that different issues arising during the process would be directed to and discussed with the respective organizations. This led to a logical distinction between the different parts of the application that were implemented mostly sequentially. The reliability of the system was ensured by conducting multiple repetitive tests, both with static images and video stream.

3) Constraints: The constraints imposed on the development of this application can be divided into timing and technical ones. The project had a time-line of approximately four months (May 2016 - August 2016). Even though a general description of the desired system was known from early on, further technical details and information for each step were acquired during multiple meetings with the stakeholders of the SRF2 project. On one hand this was a good way of focusing only on a few things at a time and organizing a logical sequence of the work that had to be done. On the other hand, not having an overall picture of all the steps needed from the beginning would occasionally not leave enough time and room for more well thought design choices. The technical constraints regarding the implementation, are considered to be the use of GStreamer for the streaming of media and the MQTT protocol in order to receive information about the on-sight vehicles from the surrounding environment.

B. Tools

This section will describe the tools and plug-ins used for implementing the application, as well as relevant implementation details. This information is divided into subsections in order to accurately address each part.

1) Operating System: Due to the fact that GStreamer uses google's gold linker, which is not included in the android ndk for windows [24], the application was not possible to

TABLE II
ANDROID FRAMEWORK

IDE	Plugin	Android SDK version	Android NDK version
Eclipse Neon	andmore plugin for Eclipse (successor of ADT)	r24.4.1	r12b
Eclipse Juno	ADT plugin for Eclipse	r24.4.1	r12b
Android Studio	-	r24.4.1	r9d

build on windows. Following the guidelines from the official GStreamer documentation that suggest the replacement of a specific android toolchain file with an older version, did not solve the issue. A 16.04 version of Ubuntu was used instead for developing the TrustMe application and it proved smoother for this purpose.

2) IDEs: Since the GStreamer documentation recommends the use of Eclipse for building and running the tutorials, that ended up being the default option in this case, as well as the version delivered to the stakeholders. The online examples are created for the purpose of working ”out-of-the-box” with Eclipse, which is not the case with Android Studio. However, in order to be able to extract useful information regarding the development process and due to the fact that Android Studio is the official IDE tailored for android development, the application was implemented with both IDEs. Table II illustrates the core differences between the IDEs regarding plug-ins, android SDK and android NDK versions. As indicated by the table, the latest NDK version did not work with android studio at the time of testing, hence some investigation was needed in order to find a version compatible for building GStreamer. The biggest challenge in using android studio was the need to setup a custom build.gradle file in order to properly read the GStreamer libraries. The purpose of this file was to pack those libraries into .jar so as to be included in the application. It is in this file also that the automatic NDK-build call has to be disabled. This allows for the use of a custom Android.mk file which is essential for building a GStreamer application.

3) Libraries: The specific gstreamer version used in this project is the gstreamer-1.0-android. The custom Android.mk file mentioned earlier, includes elements necessary in order to utilize the GStreamer library: the GStreamer root folder path, the GStreamer NDK build path and the GStreamer plug-ins required for the intended functionality of the application.

In order to implement the MQTT protocol functionality, the mqttv3-1.1.0.jar library was used. This library provided the means for an implementation of a publish-subscribe messaging functionality that ran as an asynchronous task.

C. Proof of concept (POC)

Considering the functional requirements, quality attributes and constraints established for this project, the application consisted a successful proof of concept. In essence, the artifact served to realize the proposed architectural solution and its feasibility and was not considered an early version of a final product.

V. DISCUSSION

This section discusses the outcome of this research. This includes the results of the study in relation to the methodology followed, the feasibility of the proposed architecture and the way of working in disjoint teams. Answers to the research questions and suggestions on future research are also part of this section.

A. Process

The DSRM framework proved to be an appropriate methodology to be utilized in order to prove the feasibility of the proposed architectural solution. DSRM provided a structure for the implementation of the TrustMe scenario while it assisted in gaining useful insights regarding the development process, from a developer's perspective. Consequently, in this research the process of implementing the artifact is thought to have provided useful information about the overall way of working in a self-organizing team, as well as evaluated certain practices adopted by other parties involved.

From the perspective of a junior member introduced to the task of implementing the TrustMe scenario, it was clear that in some steps during the development of the application there was not enough information available or some of the information was not as straightforward as expected. The slight offset of the shapes drawn on screen, the threat level as referred to in the proposed architecture, as well as a visual representation of the scenario depicting more than one shape identifying one threat, are considered minor deviations between the initial design and the artifact.

Even though the above deviations did not impact the purpose of the application, they consisted the basis to investigate the reasons behind them. These reasons consist the answer to RQ1 and are thought to be accurately represented by the following:

1) Lack of information: Due to the fact that the different parts of the application were implemented mainly sequentially, access to new information was also acquired in a similar way and thus it was not readily available. Furthermore, in some cases during the development process, it seemed that information about aspects of the application's design was missing. It is therefore thought that certain areas could have been investigated more beforehand. For instance, the step of transforming world to screen coordinates did not prove as straightforward as initially suggested, hence there might have been a misconception among the members of the SRF2 team about the effort and time needed in order for a developer to achieve the intended functionality. More information gathered on the subject at the planning stage and during the interviews performed as part of practicing the extended SARAD approach, could have prevented issues arising during the development. However, since the team members are mostly working within different areas of expertise, it is considered challenging to identify where more information might be needed later on during the development.

Majchrzak et al. discuss the problem of knowledge integration within cross-functional teams, concluding that an extensive dialogue among the different participants is not necessary in order to overcome that problem [6]. Instead, the authors suggest a set of practices to be followed by the team in order to narrow down the knowledge gap. However, the above implies the active participation of all team members during the process, which may not be feasible at all times.

2) Lack of planning: As lack of planning in this case it is meant the lack of initial structure and information about all the different parts that consisted the application. That being said, the requirements as well as the desired quality attributes and constraints were set from early on. In essence, it was a specific plan that was missing or was not documented. As a result, most information was acquired unofficially through discussions and meetings. One could argue that planning the design and structure of an application is solely the job of the developer implementing the software. However, breaking down all the intended functionality of the software and creating a mind-map is considered necessary, especially when a project spans across multiple different organizations.

It is worth noting that the lack of information and planning does not refer to potential false assumptions in the proposed architecture but rather to a result of the team setup that this project followed. Moreover, this information is thought to be mainly derived from the process of implementing the software and is not considered obtainable otherwise. Thus, this research proposes suggestions on further optimizing the suggested framework in order to bridge the existing gap between the initial design and the implementation.

B. Suggestions

Taking into account the results of the current research and the answer to RQ1, it is thought that there is enough information at this point for a discussion on potential suggestions regarding the extended SARAD approach. Since this study refers to the context of working in self-organizing disjoint teams, it is considered appropriate that the discussion covers the following two aspects:

1) Work sequentially: In the case of the TrustMe scenario, the work regarding the proposed architectural solution and the creation of the artifact were done sequentially. Moreover, at the time the refined architecture was introduced, the assumption was that all developers were present during the interviews and the final workshop. However, that plan was altered afterwards as the implementation of the TrustMe application became part of this research. Not participating in the process of refining the architecture, resulted in the need to quickly acquire most of the necessary information in order to develop the application. At different stages during the process the information provided by the team was slightly at a higher level and a bit general. Due to the above and since it may not be considered realistic to expect that a developer would always have access to all aspects of defining a system's architecture, this research suggests the creation of a mind-map as a step preceding the iteration of interviews introduced by Sundgren. The aforementioned mind-map would in this case be a division of the different aspects of the application, similar to the one described in the design and development step of DSRM in the research methodology section. Identifying the multiple aspects of the application this way would provide more awareness regarding the resources and effort needed in order to implement each part, as well as influence the interviews towards a focus on all areas that are part of the application. As a result this would provide any new member with a structure of what needs to be implemented and in turn make gathering information from the different organizations easier and more efficient. Figure 7 provides an updated representation of the SARAD approach and its extension, with the addition of a proposed mind-map, in this case for the TrustMe application.

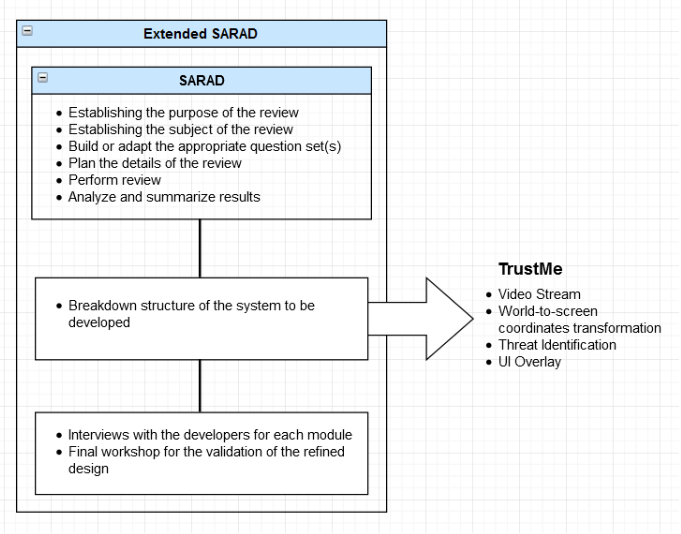
Extended SARAD
SARAD
Establishing the purpose of the review
Establishing the subject of the review
Build or adapt the appropriate question set(s)
Plan the details of the review
Perform review
Analyze and summarize results
Breakdown structure of the system to be developed
Interviews with the developers for each module
Final workshop for the validation of the refined design
TrustMe
Video Stream
World-to-screen coordinates transformation
Threat Identification
UI Overlay

Fig. 7. The SARAD approach and its extension with the addition of a mind-map for the TrustMe scenario

As mentioned at an earlier section, the challenges encountered did not impact the purpose of the TrustMe application. This however could be attributed to the fact that the application was relatively small in size, or that individual efforts led to successfully combining all pieces together in order to create a functioning artifact. In any case, one has to eliminate any indication of subjectivity as much as possible. In addition, the suggestions do not only refer to a project similar to the TrustMe scenario but aim at potentially scaling the solution to larger projects. Having said that, a large project that consists of multiple different parts is considered more risky, hence any information that goes unnoticed or any false assumptions made early on, will have a greater impact on the final artifact. It is therefore suggested to make use of a mind-map in order to tackle such issues.

2) Work in parallel: An alternative to working sequentially would be that the extended SARAD approach is applied in parallel with the development process. This essentially means that each identified module of the system to be developed would be subjected to an iteration of interviews, refined design, implementation and evaluation. In case problems arise or the evaluation results in an insufficient outcome for the desired functionality, the iteration would be repeated taking into account the problems that arose during the process. An important advantage of this method is thought to be the overall time and effort spent on the project. For instance, the required information for each step of the development process would be acquired during each iteration and thus unnecessary repetition of steps in order to spread the information would be eliminated, speeding up the overall time spent on the various tasks. Moreover, considering the fact that the junior architect in this case also acted as a coordinator of redefining the architecture of the project [8], it is only reasonable to assume that he acquired overall knowledge and understanding about the different modules of the system rather than domain-only knowledge. A combination of the architect's work with the developer of the application would in this case enhance the process and the design of the application. That being said, some of the design choices would potentially differ.

C. Contribution and future work

The objective of this research was to investigate the feasibility of a proposed architectural solution resulting from an extension of the SARAD approach. In order to achieve that, the DSRM was utilized as the default methodology. The results of the study prove that the proposed architecture was indeed feasible. In addition, the study suggests improvements in order to enhance the extended SARAD approach for use in similar or larger projects.

Considering the above, the research at this stage is thought to be contributing in two different ways. By designing and implementing an artifact based on the suggested architecture, it essentially validates in a way the author's methods and practices leading to that architectural solution. Moreover, the suggestions deriving from the implementation can potentially be utilized to form a complete methodology that will assist in the development of larger projects.

VI. Acknowledgements

I would like to thank my supervisor, Håkan Burden, for introducing me to the challenge of implementing the TrustMe application, as well as his patience, motivation and assistance throughout this research.

References

[1] R. Hoda, J. Noble and S. Marshall, ”Organizing self-organizing teams,” in *ICSE '10 Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, Cape Town, South Africa, 2010, pp. 285-294.

[2] M. Fowler and J. Highsmith, ”The agile manifesto,” *Software Development*, vol.9, no.8 , pp. 29-30, 2001.

[3] A. Begel and N. Nagappan, ”Usage and perceptions of agile software development in an industrial context: An exploratory study,” in *ESEM '07 Proceedings of the First International Symposium on Empirical Software Engineering and Measurement*, 2007, pp. 255-264.

[4] R. Hoda, N. Salleh, J. Grundy and H. Tee, ”Systematic literature reviews in agile software development: A tertiary study,” *Information and Software Technology*, vol. 85, pp. 60-70, 2017.

[5] R. Hoda and L. Murugesan, ”Multi-level agile project management challenges: A self-organizing team perspective,” *Journal of Systems and Software*, vol. 117, pp. 245-257, 2016.

[6] A. Majchrzak, P. More and S. Faraj, ”Transcending Knowledge Differences in Cross-Functional Teams,” *Organization Science*, vol. 23, no. 4, pp. 951-970, 2012.

[7] R. Nord, P. Clements, D. Emery and R. Hilliard, ”A structured approach for reviewing architecture documentation,” CMU/SEI- 2009-TN-0302009, SEI-CMU, Tech. Rep., 2009.

[8] E. Sundgren, ”A structured approach to review and refine architecture documentation,” Bachelor Thesis, Department of Computer Science and Engineering, University of Gothenburg, Gothenburg, March 2016.

[9] E. Wallgren. (2013). *Second Road fas 2* [Online]. Available: https://www.vinnova.se/p/second-road-fas-2/

[10] E. Woods and N. Rozanski, ”Unifying software architecture with its implementation”, in *ECSA '10 Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*, Copenhagen, Denmark, 2010, pp. 55-58.

[11] G. C. Murphy, D. Notkin and K. J. Sullivan, ”Software reflexion models: Bridging the gap between design and implementation,” in *IEEE Transactions on Software Engineering*, vol. 27, no. 4, pp. 364-380, 2001.

[12] dSpace Scalexio. (2017). *Scalexio* [Online]. Available: https://www.dspace.com/en/ltd/home/products/hw/simulator_hardware/scalexio.cfm

[13] AGA Project. (2014) [Online]. Available: https://developer.lindholmen.se/redmine/projects/aga/wiki

[14] D. North, (2006). *Introducing BDD* [Online]. Available: http://dannorth.net/introducing-bdd.

[15] C. Solis and X. Wang, ”A Study of the Characteristics of Behaviour Driven Development,” in *SEAA '11 Proceedings of the 2011 37th EUROMICRO Conference on Software Engineering and Advanced Applications*, 2011, pp. 383-387.

[16] GStreamer. (2015). *Gstreamer framework* [Online]. Available: https://gstreamer.freedesktop.org/

[17] Oasis Open. (2015). *Mqtt Version 3.1.1 Plus Errata 01* [Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.pdf

[18] K. Peffers, T. Tuunanen, M. Rothenberger and S. Chatterjee, ”A Design Science Research Methodology for Information Systems Research,” *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45-77, 2007.

[19] M. P. Uysal, ”Towards a Software Engineering Research Framework: Extending Design Science Research,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 3, no. 2, pp. 22-26, 2016.

[20] K. A. Piirainen and R. O. Briggs, ”Design theory in practice - Making design science research more transparent,” in *Service-Oriented Perspectives in Design Science Research - 6th International Conference*, pp. 47-61, 2011.

[21] J. Dul and T. Hak, *Case study methodology in business research*. Great Britain: Elsevier Ltd, 2008.

[22] L. Bass, P. Clements and R. Kazman, *Software architecture in practice*, 3rd ed. Addison-Wesley Proffesional, 2012.

[23] Android NDK [Online]. Available: https://developer.android.com/ndk/index.html

[24] GStreamer documentation. (2013). *Gstreamer SDK home* [Online]. Available: https://gstreamer.freedesktop.org/documentation/