Verification of Access Control Policies in Software Architectures
Proposal

Proposal of

Julian Hinrichs

at the Department of Informatics
Institute for Program Structures and Data Organization (IPD)

Reviewer:	Prof. Dr. Ralf H. Reussner
Second reviewer:	Prof. Jun.-Prof. Dr.-Ing. Anne Koziolek
Advisor:	M.Sc. Stephan Seifermann

14. Mai 2018 – 14. September 2018

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

PLACE, DATE

. . . . . .

(Julian Hinrichs)

Abstract

English abstract.

Zusammenfassung

Deutsche Zusammenfassung

Contents

List of Figures

List of Tables

1. Introduction

Nowadays, systems are more complex and connected. Therefore security aspects get more important. By virtue of recent events, users are more aware about their personal data. They want to know how the system they are using processes and stores their data. Therefore, non-functional requirements become more important. General examples for non-functional requirements are law regulations, performance or data transperency. Data transperency in this context means, transperency on how personal data is processed inside the system.
In my upcoming thesis, I will focus on data protection, a more specific case of law regulations, and data transparency. Both non-functional requirements are well reflected on access control. Therefore, I will identify and analyse dataflows to make sure access control policies aren't violated.
I will work on the architectural level. Furthermore, I will use the Palladio Component Model (PCM) as my architectural descripton language (ADL). Currently, in PCM dataflows aren't a first level entity. So the architecture and dataflows are documented seperately. In the current state some problems arise. One of the main problems is that consistency isn't ensured between the documented dataflows and the design model. Another aspect is that security charateristics often evaluated inside the code. If flaws in the architecture model are detected inside the code, it is expensive to fix them. Expensive in this context means, you have to allocate resources (manpower, time, money, etc) to this task. The problem I try to address it that the early stages of the design phase (in the waterfall model) arent't covered sufficinetly.
As a solution, Stephan Seifermann [2] proposed to extend the PCM-language to include dataflows on the architectural model. This allows to have analyses on the architectural level. I chose PCM-language as my ADL, because the PCM offers more than only modelling the architecture. With PCM, I am able to conduct different anylayse techniques. These techniques include predictions for cost, reliability, maintainbility and performance. Also simulations for the designed system are avaible.
In my upcoming thesis, I will use CoCoME. The system is more detailed explained in section 2.2. For now it is sufficient to know that CoCoME is a system which abstracts the inventoryman-agement of a big supermarket like Lidl or Aldi.
In CoCoME, I am going to define constraints to ensure non-functional requirements like data protection. After that, I am identifying dataflows inside the system. By using the constraints and the dataflows, I will use logic programming to solve the resulting constraint system. This result, I am going to use and analyse.

2. Basics

2.1. Componentbased systems

Componentbased systems consists of, as the name states, components. A component is defined as unit, which may provide or require interfaces. For clarification, in big systems you have components handling the access to a database, components for the UI and so on. The different components communicate with each other via predefined Interfaces. Note that a component-based system is modular, which means you can easily swap components in and out. Often it is possible to reuse design know-how from other components.
In the next section, I will give a brief overview over CoCoME, the system I am going to work on and a big part of my upcoming thesis.

2.2. CoCoME

CoCoMe is a componentbased System, which ~~are~~ abstracts the inventory-management and selling process of a big vendor like Lidl, Aldi or some others. This system is componentbased (2.1). It consists ~~out~~ of many components, which handle the differnet Use-Cases for CoCoME. THe following list is a quick overview of what CoCoMe is (currently) able to achieve.

- Selling goods with express checkout
- Order products for the inventory and review the ordered products.
- adding different services through an API.
- Connectiong to a database

CoCoME exist in various variants, which were developed over the time.The codebase, used technologies and the deployment may vary on each variant. Likewise for some variant there are also evolutionary scenarios that may change the deployment of CoCoME or adding additional technologies to the system.For further reading on CoCoME please see [1].
For my thesis, I will look closely in Hybrid Cloud-based Variant with the addition of a Pickup-shop. The deployment and some mentionable aspects will be covered in the following.

2.2.1. Archtitectural overview

First of all, I will take a look on the architectural view of CoCoME. This can be seen in subsection 2.2.1.

The fundamental layout of the CoCoMe, which I am using, is deployed in different layers. On top there is a layer for the frontend, an layer for webservices and a final layer for the program

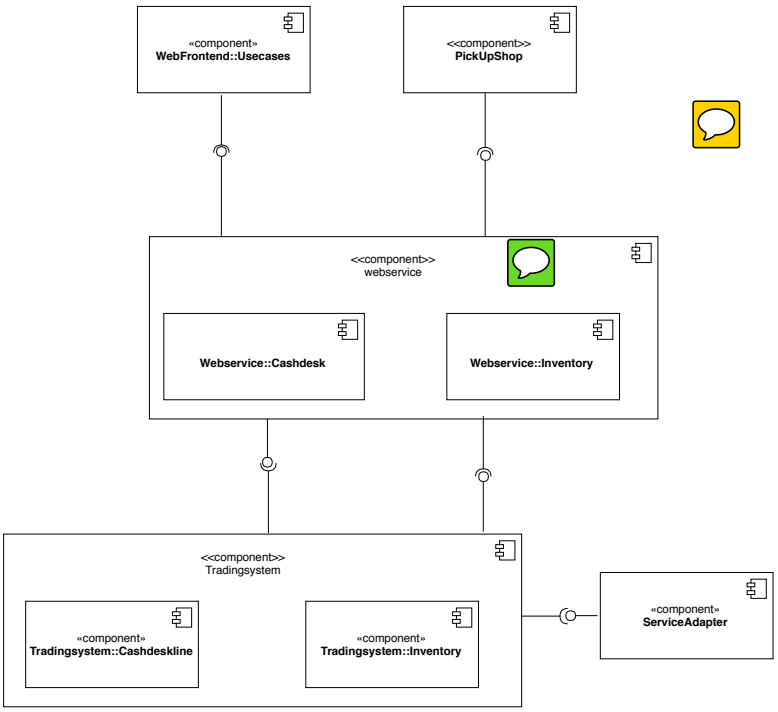

Figure 2.1.: Architectural overview over CoCoME

logic. The frontend layer is used to have differnet frontends for customer access.
The webservice layer is used to add different services, like analytics, to the system.
The programm logic layer holds the basic logic, the tradingsystem, of CoCoME. This tradingsystem divides into two parts, the cashdeskline and the inventory component. The cashdeskline handles the various user inputs. This inludes the different products an user purchased, the payment method and the displaying actions on the UI. The inventory component handles the connection to the underlying databases and the consitency of the stock items.

2.2.1.1. Adding a PickUp-shop to CoCoME

The concept of a PickUp-shop is, that a customer orders goods online and pay them directly or pay at the PickUp-shop . The purchased goods are provided by the chosen PickUp-shop, where the customer collects them. In this scenario CoCoME changes from a closed system (finite employees access from finite locations) to an open system (user may connect from all over the world). This system may be deployed on one or various servers.

2.2.1.2. Database Migration

In this case the CoCoME system deploys the database in a cloud-environment. The purpose of the decison is, if CoCoME grows in user numbers that the scalability of the current cloud isn't sufficient enough anymore and more CPU power is needed. Note that, this change of the cloud environment can be done during the runtime.

2.2.1.3. Adding a Service Adapter

The service adapter is a technique to abstract the database access. Its mainly used the not be reliant to the layout of the underlying database by adding a layer of abstration

2.3. Definitions

In this section I will give a definition of some important terms and concepts which are widly used inside this proposal and the upcoming thesis.

2.3.1. Confidentiality

Confidentiality is prsent, if there is no unauthorised informatio retrieval possible.

2.3.2. Data protection

Ability of a real person to control the flow of his/her personal data in a given system.

2.3.3. Dataflow

One can imagine a dataflow as the journey of information through a system. More precisely, a dataflow is how information are altered by process until the reach their final station.
The two figures below will clarify this definition:

2.3.4. Constraints

Constraints define limits for a system or in this case, a dataflow.

2.4. Palladio Component Model

The PCM allows more than only modelling the systems architecture. It's also possible to analyse the modeled system regarding performance, avaibility . The PCM defines four roles in the development process. These are the component developer, software architect, system deployer and the domain expert. Each role creates and use different models to fulfill their task. In the following, I will give a quick overview over the four roles, mainly which models are created by each specific role.

2.4.1. Component developer

2.4.2. Software architect

2.4.3. System deployer

2.4.4. Domain expert

3. Concept

In this section, I will show and explain the basic concept of my upcoming thesis. This will cover the ~~the~~ main question I am going to answer, the methods, I am going to use and the main goal I plan to achieve. Further, I will show and explain an example to capture the basic idea of my thesis.

3.1. Basic Concept

In general, I want to address the problem, that the early design phase is not well covered, considering security requirements. Often security is evaluated inside the code. I want to propose a solution, that both datflows and the architecture of the system are documented in the same model. Through this solution it is possible to evaluate security, especially access control, before a line of code is written.

A goal I want to achieve, is to extend the PCM-language to include dataflows in the architecture model. I want to address the issue that currently, in PCM, dataflow models and the architecture model are documented seperately. This also addresses the issue that inconsistency between he models. This inconsistency may cause errors in the system regarding non-functional requirements.This allows that changes in the architecture model will be also in the dataflow models.

To clarify the goal for me and the reader, I thought about a clear questions I want to solve. :

.

For the question to answer, I have to measure their impact. I will do this on an example system, CoCoME. I explained CoCoME in section 2.2. In the following, I will explain how I am going to conduct the case study on CoCoME.
First, I am going to present a scenario, then I am going to talk about the problems within this scenario and finally I will evaluate the scenario.

3.2. Scenario

The scenario I am going to use, is the Use Case 13 (UC13) specified in Technical Report [1] . In this use case states that the stockmanager is able to retrieve specific data concerning a specific customer. This specific data are the purchased products by the customer.
Currently the Technical Report [1] only states, the ID of a customer is given to an authenticated stockmanager. Then the stockmanager is able to retrieve all the purchased products of the specific customer, assimilated to a report. This usecase may be implemented in different ways. In general, I thought about two different implementations.

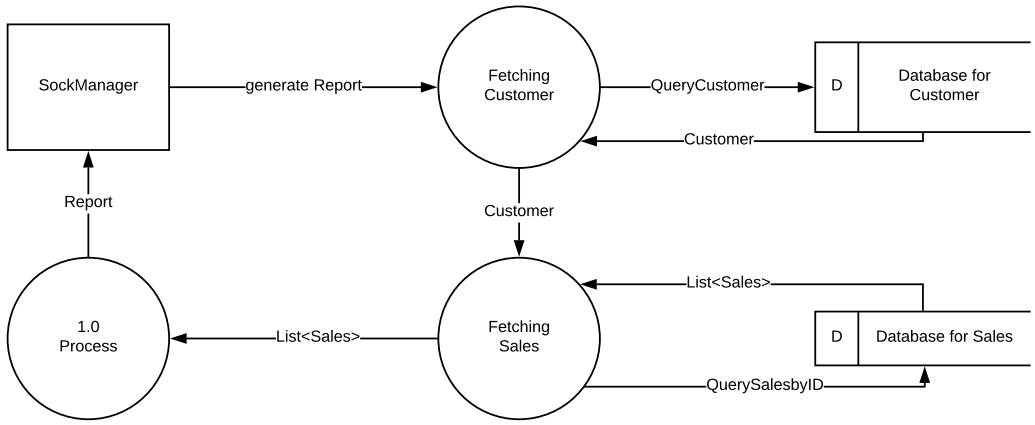

Figure 3.1.: Dataflow for the implementation using a list in the customer object

- **To the sales class an ID is added and to the customer class a list, which contains all ID's of his/her purchases**
  An ID is added to the Sale class and the customer class get an list added in which all the IDs of the conducted sales are stored. To implement the Use-Case, the system has to query each ID in the list to the underlying database and build the report out of that.

- **Store saves the products purchased and connect them to the customer via the ID**
  In this case, the sale class gets a parameter which identifies the customer who conducted the sale. To implement this USe-Case, the system queries the underlying Database for all sales and the extract the sales with the corresponding ID. Then the report is build.

The general dataflow of this scenario is depicted in section 3.2.

The stockmanager gives the ID of the customer as parameter in the system. Then a query is executed to retrieve the sales in which the customer participated. Each sale holds the involved products. Out of the retrieved sales, a report of the purchased products is generated and provided to the stockmanager.

In the dataflows that is shown is the dataflow in CoCoME for my chosen implementation. Note that queries to two different databases are made. This is because, the current deployment of CoCoME have different servers for the stores (where the sales are stored) and the web application (where the customer information is stored).

In this particular scenario there are different flaws. Regarding the data protection, the stock-manager may obtain inforamtion his role isn't allowed to. Also, some personal information may be extracted from the created report. A simple example for this flaw is, if a customer buys a lot of baby products, s/he became a parent recently. Such information is worth protecting. Also the principle of data poverty is violated. The stockmanager doesn't need the personal informations to fulfill her/hs task inside the store.
I want to investigate different views. these views are:

-

3.3. Evaluation

4. Organisation

4.1. Shedule

The next figure shows a rough overview over my schedule in the thesis.

4.2. Riskmanagement

In this section I will focus on risk management of my thesis. Therefore, I will point out some aspects, that may cause delay of my work.

- **Roles in CoCoME**
  Currently a rolemanagement is not implemented in CoCoME. I will add this. For this I may use more time than I expected.
- **Implementation of Scenarios**
  Some of the scenarios I am going to use, aren't par tof CoCoME yet. It may take more time than expected to implement these.
- **Extension of PCM**
  Currently PCM doesn't support dataflows in the architecture model. It may cost more time than anticipated to extend the PCM.

Bibliography

[1] Robert Heinrich, Kiana Rostami, and Ralf Reussner. “The CoCoME Platform for Collaborative Empirical Research on Information System Evolution”. In: (2016).

[2] Stephan Seifermann. “Architectural Data Flow Analysis”. In: *13th Working IEEE/IFIP Conference on Software Architecture, WICSA 2016, Venice, Italy, April 5-8, 2016*. 2016, pp. 270–271. DOI: 10.1109/WICSA.2016.49. URL: https://doi.org/10.1109/WICSA.2016.49.

A. Appendix

A.1. First Appendix Section

Figure A.1.: A figure

…