Component Identification in Software Configuration Management Applied to the Development of Framework Traceability

Denise do Rocio Maciel[1], Cláudio Roberto Agner[2], Luma Lopes[3], Maria Salete Marcon Gomes Vaz[4], Mônica Cristine Scherer Vaz[5]

[1,2,3,4]Postgraduate Program in Applied Computing, State University of Ponta Grossa, Brazil
[5]Postgraduate Program in Production Engineering, Federal Technological University of Paraná, Brazil

Abstract— *Traceability in the agricultural production chain enables us to identify the origin and the process by which a product was subjected to its availability to the end consumer. To develop a framework for traceability of the productive grain process, specification and implementation was performed by following the Software Engineering standards, with the management of Software Configuration. This article presents the first phase of this administration, component identification, applied to the development of this framework, as a team, with limited human and financial resources. As a result, the approach has allowed greater control of software components, the assimilation of the importance of team work and greater independence between members of the project. To reduce the development time of new projects, one solution is to establish greater granularity of components to be managed and define a part of the team responsible for the Software Configuration Management*

Keywords— Agricultural Production, Managing Software Configuration, Software Development, Traceability.

I. INTRODUCTION

Traceability in the agricultural production chain enables us to identify the origin and the process by which a product was subjected to its availability to the end consumer. The traceability techniques are applied for quality control and to preserve the identity of a product, with monitoring and management of the productive process phases. The impact caused by problems in the production process can be minimized with an effective traceability process, since it reduces the time between the occurrence of the problem and identifies the source, prevents the recurrence, decline in production, lack of quality and, consequently, increased costs.

The requirements required by the regulations and quality standards for meeting the criteria of food security, microbiological analyzes of foods, good agricultural practices and tracing to identify the origin of the product, it becomes necessary for accredited laboratories, sanitary inspection system and quality certifications. In this process, the certification aims to provide the buyer or user of the product a guarantee compliance to standards or technical specifications established (CONCEPTION, 2005). When a company certifies its product, it assumes that the information you are providing is important to consumers and that they will respond by changing their consumption decisions.

According to Eckschmidt et al. (2009), some participants are critical for a company to evolve in a tracking process, other than agents of the productive chain and consumers. They are: (i) Regulatory Agents - components that define the rules, standards and laws to be followed; (ii) Facilitator Agents - companies that provide services and offer products to support the process of tracking, as defined by regulatory agents; (iii) Certifying Agents - components that evaluate and certify that the producer is fulfilling the established traceability requirements.

In Vaz (2014) the RastroGrão Framework was specified, with the aim of customization of the production chain for any type of grain, aiming to meet the demands for traceability of the production process, according to each agent in the chain. This framework was developed with guidelines outlined in the Software Engineering (WAZLAWICK, 2013; Pressman, 2011; DUNCAN, 1996), where Project Management, Time Management, Software Configuration Management, Quality Management, Document Management, Content Management and Knowledge Management were implemented in the search.

The framework allows the configuration of the agricultural production chain and grain traceability. The Resolution no. 748/2014 (SESA, 2014), of the State of Paraná, constitutes the need for improvement of the framework, allowing the traceability of any vegetables, tied to quality techniques, specifically, management of software configuration. Through the Tracked Food Program, this resolution has determined, from June/2016, the obligation of labeling with traceability information.

Cunha et al. (2004) reported that, despite efforts to improve the quality in software development, and consequently increase the degree of success in software projects, there is the difficulty of organizing adopted concepts and practices related to configuration management. The deployment requires investment both in terms of financial and human resources, and these resources are scarce, mainly in small organizations.

The objective of this paper is to present the identification of Software Components, in Configuration Management, applied to the development of the traceability framework, using open source tools and operating in a scenario of resource optimization. The identification of the components acts as a facilitator for the deployment of configuration management in companies or other applications.

Therefore, this article is structured, beyond this introductory section, as follows. In Section 2 the state of the art of traceability and traceability framework are described. In Section 3 Software Configuration Management is approached. In Section 4 we present the contribution of this article that specifies the implementation of the survey component of the management of software configuration applied to the process of developing the framework for traceability of grains. In Section 5 correlated work is analyzed with the advantages and disadvantages of each one. In Section 6 conclusions and prospects for future research are discussed.

II. TRACEABILITY AND FRAMEWORK TRACEABILITY

Traceability can be defined as the ability to document, historically, the application or location of products and procedures used during the production chain steps (ISO 22005, 2007). By means of traceability, it is possible to follow the flow of the product in the production chain, resulting in the identification, collection, processing and storage of records, with information and critical points in the process (DERRICK AND DILLON, 2004).

Food safety for consumers is one of the benefits of traceability, since in cases of risk, it becomes possible to withdraw products from the market, as well as the possibility of diagnosing problems and failures in all phases of production. Thus, the decision of the producer/manufacturer becomes agile, avoiding damage to the consumers (FURLANETO and MANZANO, 2010).

Vaz (2014), to create the Framework RastroGrão model, specified to assist in the tracking process, integrating all agents in the production chain, evaluated some models of traceability. The RG - Traceability of grains (Zegers, 2007), developed to evaluate a preserved identity program and traceability for wheat, documenting the entire process, from the reception, drying, storage and processing, focusing on the management of insects during storage. Since the Caderno de Campo Digital (TIBOLA, C.S.; FERNANDES, J. M.; 2009) was developed for the internet. With the goal of keeping records of provenance and adopted practices, from production to post-harvest, and how to record physical and chemical properties of the soil, planning crop rotation, soil management, and seeding, seed treatment, fertilization, weed control and application of growth regulators, monitoring and control of pests and diseases, applications of fungicides and insecticides, storage unit, drying, and thermometry and aeration of grain.

By means of its structure, it is possible to trace products, the phases of the production process of each product, as well as the data to be tracked in each phase. In accordance with the defined structure, the process can assist in traceability of any type of grain, since the system can be managed by the user according to their needs. Since the process of traceability is not static and should cover all agents in the production chain, the user can customize the process according to each need, i.e., with the rules of agribusiness, with the new rules that may arise and with the constant research in the area which may result in new needs in a traceability process.

RastroGrão was modeled with five (5) modules, as follows: 1) General Records, including the management of users that have access to the Framework, Company Management, including those with the process of structured traceability and will be the managing bodies of data and Property Management, including the locations where the grain is produced and where the traceability process parts; 2) Managing the structure of traceability, related to the management of products, phases and attributes that will be part of the library, which can be customized for each company; 3) Customization module, where each company selects the products, phases and their attributes, which will be part of its structure to be traced; 4) Record the data, module where the company will make the insertion of the production data, traceability and, 5) Consultation of data and generation of the QRCode labels, contains the information that each company entered in the database and which will be available for consultation and/or end user.

The framework has been defined with a user-friendly interface, both for the creation of the traceability structure as to the user who will do the recording of information. Access is carried out via the internet, allowing information to be accessed by all agents in the chain. This way, it is possible to meet the needs of each company, since it is not a system with predefined, but yes, customizable and easy to adapt to the rules of each agribusiness.

III. INDENTATIONS AND EQUATIONS

The software configuration management is the area that indicates how different versions of the artifacts involved in software development should be modified and labeled (WAZLAWICK, 2013).

According to Pressman (2011), the management of software configuration can be understood as a set of tasks developed throughout the quality management, with the aim of managing change through the entire life cycle of software. It is possible to identify artifacts that collectively define the software configuration, manage changes in one or more of these artifacts, facilitate the construction of multiple versions of an application and shall ensure that the quality of software is maintained as the configuration evolves with time.

The management of Software Configuration include the concepts that follow (PRESSMAN, 2011; WAZLAWICK, 2013):

- **Software Configuration Item:** Is the component of development that will be controlled by the management system. When the component is considered an item, it can only be changed through definition of change set out in the development plan for the configuration management. The items can be basic or compound. The basics are formed by an object, since the compounds can be formed by other basic items and compounds. For example, document, software tools, set of test cases or program component with a name.
- ***Baseline***: It is marked by the provision of one or more configuration items approved in consequence of a technical review. An example of a baseline can be the set of interface requirements that have been approved by the client for development or a version of the system delivered to the customer and approved in the acceptance tests.
- ***Release:*** Is the distribution of a software version, or a configuration item, for the production environment.
- **Metamodel Document**: Available in the configuration management repository, responsible for initiating the interaction between users and the repository, and contains the storage structure of various artifacts. Specifically, defines the files storage and information forms, forms of access and data visualization software, managing security, data integrity and the ability of the current repository to be extended to meet future needs.

IV. COMPONENT IDENTIFICATION IN CONFIGURATION MANAGEMENT

The configuration management process is initiated by the identification of components, followed by the step version control, responsible for tracking the configuration items, generating historic and relationship between them; change control, which indicates the reasons which led to the implementation of changes in a software configuration item; configuration audit, which examines whether the configuration items are present in a version, baseline or release to see if they are present in the repository; and a status report, responsible for providing centralized information about the changes.

This article deals with the first stage of the software configuration management, identification of components. Wazlawick (2013) defines six attributes to be cataloged: software configuration items, relationships, versioning, software configuration and baseline and release set. Then follows the detailed definition of each of the attributes.

4.1 Identification of Software Configuration Items

Tables 1 and 2 show the components used in the management of configuration. Column CIS brings together the components in software configuration items, and if accompanied by the word "set" indicates that the configuration item is composed. All tools involved in the project are Open Source. Table 1 describes the software components. To store them, we used Repository II, made available by the tools GitHub and Git.

The choice of tools was based on the results of Palestine (2015), where the systems are analyzed according to functionality and usability requirements. In relation to the functionality the aspects of configuration were considered, acceptance and processing of data, ease of installation, security and access control. To determine the usability the notes cited by Nielsen (2010) were considered.

As a result of the study, the SVN system showed greater adequacy, followed by Git/Github and finally DokuWiki. In spite of the SVN system presenting greater credibility in the study, Git was selected due to ease of installation and for being a

distributed system, made available through the repository on Github cloud. The SVN System was used as an option to achieve integration with the data from the repository.

TABLE 1
SOFTWARE COMPONENTS AND ITEM CONFIGURATION

Software Components			
Component		**Function**	**ICS**
1.	Dropbox 3.18.1	Cloud file managing system	Repository I
2.	GitHub	Cloud encoding repository	Software set for implementing repository II
3.	Git 2.8.1	File Versioning	
4.	Tortoise SVN 2.1.0.0	Tool for visual integration of Git on Windows OS	
5.	JDK 1.8	Package of tools for developing Java applications	Software for encoding development
6.	Netbeans 8.1	IDE for the Groovy programming language	
7.	PostgreSQL 9.5.2	The Database Manager	
8.	Grais Framework versão 2.2.4	Framework for Web app development	
9.	Plugin Jquery 1.11.1	Jquery Library	
10.	Plugin Jasper 1.10.0	Report Framework. The plugin Jasper is used to generate reports produced by the Framework iReport	
11.	Plugin Audit-logging 1.1.0	Allows you to manage modifications performed in the Grails application. Such as save, edit and delete	
12.	Plugin spring-security-core:1.2.7.4	Manages application security	
13.	Plugin QRCode:0.7	Responsible for generating QRCode according to text/code/url established by the system	
14.	brModelo 2.0.0	Modeling tool	Software for modeling
15.	RastroGrão Coding	Coding for the RastroGrão framework	All coding modules
16.	Mantis Bug Tracker	Change manager software	Change manager
17.	OpenProj 1.4	Project managing software	Project Manager

The Framework RastroGrão was implemented through the programming language Groovy and from the Grails Framework, represented by the software configuration items for Development and Codification. The tool has open source license and allows agile software development. Groovy is a flexible programming language, interpreted or compiled, designed for the Java Virtual Machine Platform JVM, having integration with the Java language, with influence from Ruby, Python, Perl, Smalltalk and Java languages (judd et al., 2008).

Grails is a high productivity framework, based on the Programming Language Groovy, allowing inclusion of Java Commands. Aims for the availability of simple paths and friendly for quick development of systems for WEB 2.0 (ANSELMO, 2010). This framework is based on the MVC concept - Model, View, Control and operates through the persistence of domain classes, which can generate the scheme of the database. It differs from other frameworks because their services and classes can be inserted, automatically, through dependencies generated by the use of coding by convention (judd et al., 2008).

The framework is formed by other tools attached to its installation, with these extensions adding features like web container, databases, system builds, support for unit testing, support for integration testing, support for functional testing for the interface and plugins library. Follows a description of these components (judd et al., 2008):

- Scaffolding: Plugin used for automatic generation of creating operations, reading, updating and deletion (CRUD - create, read, upload, delete). The implementation is made from reduced code and, optionally, you can determine the automatic generation of database schema. With this feature, the developer can focus efforts on the definition of rules for developing project.

- Hibernate: Relational persistence Framework that provides the basis for GORM (Grails Object Relational Mapping).
-
- GORM: Framework for mapping relational databases, operating both in the definition of the scope of objects as in the relationship between them.
- Spring: Framework that provides an abstraction of the Java language, simplifying the interaction of the developer with the tool.
- Sitemash: Framework for design implementation and HTML web page rendering
- HSQLDB: Java database, allowing the developer to use it or opt for another solution.
- Junit: Framework used for conducting unit tests.

TABLE 2
COMPONENT DOCUMENTS AND CONFIGURATION ITEMS

Documentation Components			
Components		**Function**	**ICS**
1.	Development Standards	Metamodel Document	Metamodel Document
2.	Version Control	Presents information on baseline and release development	Version Report Control
3.	RastroGrão Timeline	*Baselines* and *releases details*, Giving in the Software Configuration Item "Version Report Control"	RastroGrão timeline report
4.	Rastro Grão Meeting Timeline	Specifies the time spent in meetings	RastroGrão meeting timeline report
5.	System Tutorial	Tutorial for end users of the RastroGrão system	RastroGrão Tutorial – End user
6.	Required Development Tutorials	Tutorial for developers. Discusses technical issues for the use of the tools outlined in the Software Configuration Item Metamodel "Document"	RastroGrão Tutorial – Developer
7.	Use Cases	Use Cases Specifics	All Use Cases
8.	Correction Requests	Specifies correction request	All Corrections Requests
9.	Change Request	Specifies change requests, they can be held by team members or the end user.	All Change Requests
10.	Meeting Minutes	Specifies meetings held both for project definition, delivery or build	All Meeting Minutes
11.	Project Opening	Sets goals and requirements for the initiation of project development	Project Opening Documents
12.	Project Delivery	Documents final project delivery	Project Closure Document
13.	Build Delivery	Documents each coding project delivery	Build Delivery Document
14.	Meeting Minutes	Meeting Minute Model	All model documents
15.	Use Cases	Use Case Model	
16.	Project Opening Document	Project opening model	
17.	Project Conclusion	Project closure model	
18.	Build Delivery	Build delivery model	
19.	Correction Request	Correction request model	
20.	Change Request	Change request model	
21.	Entity Modeling –	Entity relationship modeling document	All entity relationship

Table 2 addresses the components of documentation, and for their storage Repository I was used, provided by the Tool Dropbox. Dropbox was chosen for file storage not to compromise the limit offered by competitors, OneDrive from Microsoft and Google Docs from Google, in which the members of the project had accounts in use. For further developments its worth noting the need for analysis of storage tools, such as the storage area and the conditions of migrating to a paid plan.

4.2 Component Identification – Releases and Baselines Definitions

The project counted 71 baselines, 5 releases regarding the construction of the project and 6 releases regarding the request for amendment or change. The availability of each release was accompanied by a document called "Build Delivery", responsible for presenting the customer the specifications of the delivery. Table 3 follows the relationship of the defined releases and baselines. Through the configuration item "Report Version Control," members of the project were allowed to view details about the development of the components.

TABLE 3
BASELINES AND RELEASES RELATION

Baselines and Releases Identification	
Baseline 01	Beginning of the development environment containing the settings established in configuration management
Baseline 02 to Baseline 70	Each baseline corresponds to a Use Case, determined at the beginning of the project, or a set of one or more requests for corrections reported via the software Mantis
Release 01 to Release 05	Set of modules for Encoding - Each release is the delivery of a system module, the development of the project was determined in five modules. With each release delivery the document "Build Delivery" were included.
Release 06 to Release 11	Set of modules for Encoding - Each release is the delivery of a system module, the development of the project was determined in five modules. With each release delivery the document "Build Delivery" were included.

The RastroGrão traceability framework project had its development in a hybrid form, mixing characteristics of the agile project management software models Extreme Programming - XP and Scrum (PRESSMAN, 2011). The delivery stage of the software encoding was based on Scrum. This model addresses the need to define profiles, role assignment to each member of the team, definition of the product backlog, list of features to be implemented in the project, sprints management, development cycles which structure Scrum and completion of daily meetings called Daily Scrum (WAZLAWICK, 2013).

The delivery of coding is characterized by the definition of a release, as shown in Table 3, as suggested by the Scrum model. This delivery is the result of the process called Sprint. The definition of durability and deadline for development of the sprints occurred in the beginning of the project.

Also, the step of defining the product backlog was implemented, but with a difference, in the bibliography it is suggested to create a history to compose the backlog items, as the framework has been specified, we created use cases based on this specification. The use cases formed the list of product backlog.

4.3 Relationship between the Configuration Items, Traceability, Version and Configuration Software

The relationship between Configuration Items establishes the connection from one item to another, from a list of required resources. This control was specified via the configuration item "RastroGrão developer tutorial ". The dependency between the items have not been addressed, since the document has sequential steps for preparation and management of the environment.

Traceability is necessary to maintain consistency between the artifacts, and it has been as a traceability matrix example. Represented in Table 4, this matrix was inserted at the beginning of some documents, in order to generate the traceability of the internal content. The configuration items applied in this control are Metamodel Document, Use Cases Set, RastroGrão-developer tutorial and RastroGrão-End User tutorial.

TABLE 4:
TRACEABILITY MATRIX

Traceability Matrix			
Review	**Commentary**	**Date**	**Responsible**
0	Document Development	April 19, 2016	Cláudio Agner
1	Description about storage of correction and	April 27, 2016	Denise Maciel
2	Inclusion of the link to Github	April 30, 2016	Luma Alves Lopes

The version of a configuration item is a particular state of this item, during the development of a system (WAZLAWICK, 2013). To provide traceability between some configuration items the versioning system was used, described in Table 5, as part of the nomenclature. The configuration items controlled were: Use Cases Set, Change Request and Correction Request

TABLE 5
MODEL OF THE VERSION CONTROL DEVELOPMENT

Version	Review	Correction	Build	Commentary	Date	Hours	Responsible
1	0	0	0	Beginning version of the management system configuration	April 19, 2016	2	Cláudio Agner

The software configuration corresponds to a list of configuration items that comprise the software and their respective versions (WAZLWICK, 2013). Since the project is in the first version, the metamodel document was enough to make the specification.

V. CORRELATED WORK

Cunha et al. (2005) presented the proposal for the implementation of the configuration management process, based on a simplified version of the IDEAL model (MCFEELEY, 1996), for improvement of software process. The authors performed a case study, in the laboratories of the Department of Computer Science, at São Carlos Federal University(UFSCar), to illustrate the use of the approach. As a differential, the study of this article presents in detail the stage of configuration management, identification of components, with the activity of configuration management performed in real world scenario, in the development of the traceability framework RastroGrão.

Dantas et al. (2003) analyze the approach of configuration managing applied to the concept of "Development Based on Components". Again, the practical and applied in real world scenario is a differential of this article. The two studies present a detailed approach, but differ because it is focused on the first stage of configuration management, Component Identification, while Dantas et al. (2003) focuses on the second stage, Configuration Items Control.

It is worth noting that the approach of managing software configuration, with emphasis on component identification, is applied with the use of OpenSource Tools. The application in real world scenario has allowed analyzes of success or limitation of the proposed configuration items, in addition to monitoring all aspects regarding the acceptance of configuration management on the part of the team involved.

VI. CONCLUSION AND PROSPECT FUTURE WORK

The Component Identification stage was possible and effective to maintain control over the software, other than giving greater independence to the members of the project and emphasizing the importance of team work.

In spite of the study, addressing the first step, all the steps of managing software configuration were developed, allowing us to infer that the Component Identification may not be efficient for applications in companies with limited human resources. This statement is due to the expenditures to maintain the document management. The system had a total of 197 hours for its development, with the initial specification of 110 hours. To alleviate the problem, a solution for new projects is revising the granularity of Configuration Items and the definition of a team member responsible for the management of Software Configuration.

Difficulties in relation to team communication have occurred since the beginning of the project. In future developments it is recommended to define an official communication channels as an item of software configuration, as well as a document specifying the functions of each member.

The configuration item "Project Manager" was not effective due to operational failure, being replaced by the configuration items "RastroGrão Report Schedule " and "RastroGrão Report Schedule- Meetings". The first specifies the development of codification modules and the second addresses the completion date and duration of meetings. For the meeting document, a

suggested improvement is entering the data of tasks to be delivered. This information is currently in the records, which complicates access and use of the data.

As prospects for future work it is suggested to approach the other steps in the Software Configuration Management process and the reapplication of implementation with the improvements suggested, in order to overcome the difficulties of the time required to perform the management.

ACKNOWLEDGEMENTS

The author thanks to Comissão de Aperfeiçoamento de Pessoal do Nível Superior – CAPES for the financial support.

REFERENCES

[1] Anselmo, F. "Em Busca do Grails", Florianópolis: Visual Books Editora,2010.

[2] Brasil. Secretaria da Saúde Governo do Estado do Paraná. "Resolução nº 748, de 2014". Dispõe sobre a rotulagem de produtos hortículas *in natura* a granel e embalados, comercializados no estado do Paraná. Diário Oficial, Paraná, 17 dez. 2014. p. 83.

[3] Conceição, Júnia C., Barros, Alexandre L. M. "Certificação e Rastreabilidade no Agronegócio: Instrumentos cada vez mais necessários". Ipea. Brasília, outubro de 2005.

[4] Cunha J. R. D. D; Prado A. F; Santos A.C. "Proposta de uma Abordagem para a Implantação do Processo de Gerenciamento de Gerenciamento de Configuração de Software".2005. Congresso Internacional de Gestão da Tecnologia e Sistemas de Informação.v2.2005.São Paulo – SP. Disponível em: http://www.tecsi.fea.usp.br/pastcontecsi/index.php/contecsi/article/view/538_. Acessado em: Maio,10,2016

[5] Cunha J. R. D. D; Prado A. F; Santos A.C;Neto R. M. S. "Uma Abordagem para o Processo de Gerenciamento de Configuração de Software". Revista Eletrônica de Sistemas de Informação.v3.n1.2004. Disponível em <em:
http://www.periodicosibepes.org.br/ojs/index.php/reinfo/article/view/145/37>. Acessado em: 10 de Maio de 2016.

[6] Dantas, C.R; Oliveira, H; Murta L.G.P; Werner C.M.L. "Um Estudo sobre Gerência de Configuração de Software aplicada ao Desenvolvimento Baseado em Componentes". Terceiro Workshop de Desenvolvimento Baseado em Componentes.2003. *D*isponível em: <http://reuse.cos.ufrjbr/prometeus/ publicacoes/wdbc03-gcs.pdf >. Acessado em: 10 de Maio de 2016.

[7] Derrick, S.; Dillon, M. "A Guidi to Traceability within the Fish Industry". Sippo, Eurofish, 2004.

[8] Ducan, W. R. "A guide to the project management body of knowledge", 1996.

[9] Eckschmidt, Thomas., et al. "O Livro Verde de Rastreamento: conceitos e desafios". 1a. edição. São Paulo. Livraria Varela, 2009

[10] Furlaneto, F. P. B.; Manzano, L. M. "Agricultura de precisão e a rastreabilidade de produtos agrícolas".2010 Disponível em: <http://www.infobibos.com/Artigos/2010_2/AgriculturaPrecisao/index.htm>. Acesso em: 16 jun. 2016.

[11] ISO 22005. "Traceability in the feed and food chain -- General principles and basic requirements for system design and implementation".2007. Disponível em: < https://www.iso.org/obp/ ui/#iso:std:iso:22005:ed-1:v1:en>. Acesso em: 16 jun. 2016

[12] Judder C.M; NUSAIRAT, J. F; SHINGLER, J. Beginning Groovy and Grails: From Novice to Professional. Apress, 2008.

[13] Mcfeeley, R. IDEAL: "A user guide for softwareprocess improvement. Relatório Técnico CMU/SEI-96-HB-001", Software Engineering Institute, 1996.

[14] Nielsen, J. Usability 101: "Introduction to Usability". Disponível em: <http://www.useit. com/alertbox /20 3 0825.html> . Acesso em: 08 jul. 2016.

[15] Palestino,C.M.C. "Estudo de Tecnologias de Controle de Versões de Software", 2015.

[16] Pressman, R.S. "Engenharia de Software – Uma abordagem professional -7ª Edição", São Paulo: AMGH Editora Ltda,2011.

[17] Vaz, M. C. S. "Especificação de um Framework para Rastreabilidade da Cadeia Produtiva de Grãos". 87f. Dissertação de Mestrado em Computação Aplicada, Universidade Estadual de Ponta Grossa, PR, 2014.

[18] Wazlawick R.S. "Engenharia de Software – Conceitos e Práticas" .Rio de Janeiro: ELSEVIER, 2013.