Using Atlassian Tools for Efficient Requirements Management

An industrial case study

Luc Filion, Nicolas Daviot
Nuum Solutions
304-4529 Clark St.
Montréal, QC, H2T 2T3, CANADA
luc.filion@nuumsolutions.com

Jean-Philippe Le Bel, Marc Gagnon
ISC Applied Systems
290, avenue Labrosse
Pointe-Claire, QC, H9R 6R6, CANADA

***Abstract*— This paper describes an industrial case study using Atlassian JIRA® and third party plugins for requirements management in the field of transit systems. The solution presented shows efficiency in supporting the management of requirements, traceability and the systems engineering processes globally. After a short description of the technologies in action and a brief overview of the process we targeted, we describe the collection of methods and technologies put into place and demonstrate their use in order to achieve our goal: reaching CMMI-2 for requirements management and traceability. We describe the current background and scope of work, the needs and goals to meet, and then we explain our configurations and possibilities. Our results describe several sets of metrics we have obtained for traceability, starting from a customer requirement down to tests, considering multi-level analysis with specific criteria (verified, tested and justified).**

Keywords—requirements management, traceability, JIRA, Atlassian, CMMI

I. INTRODUCTION

ISC Applied Systems (ISC) is a Canadian company and supplier of Passenger Information Security and Communication Systems to the public mass transit sector. Nuum Solutions (Nuum) is a Canadian services company working in software and systems engineering, processes and methods and is a specialist with Atlassian technologies. Both entities worked together in order to meet CMMI-2 [1] level for requirements management and traceability. Achieving CMMI-2 is quite challenging and requires a healthy dose of rigour, and companies working in the field of transit and transportation systems are often required to meet such standards or their equivalents. Managing requirements and generating traceability is not a new topic. Several techniques and variations exist and have been thoroughly described as in Pohl [2]. Moreover, numerous more traditional tools exist, such as DOORS and RequisitePro [3]. Newer generations such as Jama or several dozen other native or web-based tool are either generic or specifically targeted for vertical markets. Requirement Management tools can be costly for Small and Medium-sized Enterprises (SME) and can be disconnected from other tools which are routinely used for engineering activities and operations (such as test tracking tools). Despite the wide number of tools available, many SME still manage their requirements manually (via Excel or in-house database for instance).

This paper describes a requirements management solution which is JIRA-centric. Although Atlassian JIRA does not target requirements management per se, it can be configured for managing and keeping track of requirements efficiently. JIRA is low cost and therefore affordable for SME as well as larger corporations. It can be quickly deployed and intuitively understood by systems engineers and by all other engineers, testers and managers orbiting around a system engineering project. An annual requirements management tools comparative report [4] gives JIRA the 18th position in regards with requirements management, because of many limitations. This paper describes how to overcome these limitations.

In this paper, we first describe the Atlassian technologies in use such as JIRA [5], Confluence [6], and some other third party plugins such as eazyBI [7] that we use to deploy this requirements management solution. On the technical side, we then show how we configure them in order to create the requirements management solution and to generate traceability reports.

Complementarily, we discuss about our motivations and goals for realizing this project. We describe the requirement dependency model we need to build. Then, we present some numbered results and statistics obtained using the targeted technology. Also, we briefly discuss about the competitiveness of JIRA as a tool for requirements management.

II. TECHNOLOGIES

A. Atlassian and JIRA ®

Emerging from the software industry, Atlassian JIRA is a generic work item tracker (“work items” which JIRA calls “issues”) which is widely used for tracking software bugs and tasks and is also commonly used for Agile projects. JIRA has nonetheless a high level of configurability, which allow administrators to create legitimately all types of work items. Consequently, created work items can be requirement objects of all kinds (e.g. System Requirements, Software Requirements, or

Luc Filion, et al., “Using Atlassian Tools for Efficient Requirements Management”, in 2017 IEEE International Systems Conference (SysCon 2017) Proceedings, Pages: 446 – 451. © IEEE 2017 - DOI: 10.1109/SysCon.2017.

simply Requirements). All work items have a predefined set of data fields (properties or attributes), which are common to all tracked items, such as status, priority, version, etc. These properties are also configurable, and can be altered or complemented with new custom fields which can be specifically dedicated to requirements. For instance, we may want to characterize requirements with a Requirement Type, a Mode of Operation, a Source, a development Phase, a Component, a Sub-System, and so on.

Requirements objects are therefore instantiated, characterized and tracked. JIRA has the ability to log all the changes made to an object, allowing for requirement evolutions or reporting.

One big advantage of JIRA consists in that the tracking of a requirement can be closely coupled with the tracking of everything else related to it: sub-requirements, tasks, change requests, tests, bugs, project management items, and so on. This can be very helpful to SME, the number of resources being limited. It can also be applied to lean systems engineering concepts as described in [8] [9] and practices or even Agile systems engineering [10] where the need for delivering something quickly is a live or die factor.

JIRA offers the ability to link items together. JIRA links go along with a link clause that describes the relationship between two linked items. The link clause is also configurable, and we may then correctly link requirements with, for instance, an "implements"-"implemented by" relationship. For instance, a Software Requirement may be implemented by a "Software Task" or by a "User Story" (for Agile fans). These links are the keys that enable traceability between objects (requirement to requirement, but also requirement to task, source code, test, change request, and so on).

Finally, it is possible to configure dashboards and reports in JIRA, which helps systems engineers to keep track of priorities, and to present the true progress for any given system under development.

One of the shortcomings is that JIRA does not offer built-in baselining which can be useful when analyzing requirement changes over a period of time. It also does not offer advanced features, such as branching, auditing or built-in graphical viewers, diagrams or traceability matrices. Bypasses exist for these issues. Some are addressed in this paper.

B. *Confluence®*

One problem with JIRA is the impossibility to describe requirement textually in a proper way. JIRA only provides a basic text window for descriptions and only allows to attach files (such as images or diagrams) separately. On the other hand, Atlassian Confluence compensates JIRA for these limitations. Confluence is a multi-user web-based advanced wiki editor, which offers on-line word processing and image embedding. It proposes a set of programmable macros to enhance a documents' content. Through macros, Confluence offers the possibility to connect with the JIRA database to dynamically reflect the state of requirements within the text itself. This highly reduces the manual modifications to apply to the specifications. Fig. 1 depicts such a macro.

Fig. 1. Example of a macro to be inserted into Confluence pages, which reflects the realtime state of given requirement stored in the JIRA database.

C. *Third-Party Add-ons*

By having an open and documented API for JIRA and Confluence, Atlassian opens its technology for extensions by developers or third party companies. Over time, it has created a marketplace proposing several free or low cost add-ons that are useful for requirements managements. On the other hand, some requirements management vendors have been able to hook their native tools to JIRA through APIs, and open the possibility to connect remote tools with JIRA itself.

Amongst the range of existing plugins, we used some to develop our solutions. eazyBI Reports and Charts [11] is a powerful reports and charts add-on for JIRA. It proposes a drag-and-drop tool for analyzing and visualizing predefined or custom calculated measures taken from the JIRA database. We used it in conjunction with the JIRA Misc Custom Fields add-on [12], which offers to create programmed custom fields. Finally, we used Impact for JIRA – Traceability Analysis [13] for generating requirement traceability reports.

III. MOTIVATION

Several objectives were set for this project, the final target being a CMMI-2 assessment. The first step towards this goal is to define the requirement management model to reproduce and map it in JIRA.

A. *Workitems*

In order to better meet the needs in terms of requirements management for transit systems where several engineering standards need to be followed, such as EN50155, ISC works with several levels of requirements: Customer Requirements directly emerge from the Customer's Statement of Work, System Requirements, which redefine customer requirements at a technical level. Hardware and Software Requirements define further the System Requirements. Tests are also provided to verify that requirements are met. Several levels and types of tests need to be defined as well, such as Quality Tests or Acceptance Tests. As a simple rule, we make the assumption that a passed test report confirms the fulfilment of a given requirement. Change Requests are also often applied, representing changes in the system. Changes may come from the end customer, management, engineering.

B. *Requirement met criteria*

A requirement meets its acceptance criteria if and only if all child objects are also in a met state. For instance, the Customer Requirement in Fig. 2 is not met, because of a failed production test on some dependent hardware requirements. Our JIRA configuration allows to determine such impact.

Luc Filion, et al., "Using Atlassian Tools for Efficient Requirements Management", in 2017 IEEE International Systems Conference (SysCon 2017) Proceedings, Pages: 446 – 451. © IEEE 2017 - DOI: 10.1109/SysCon.2017.

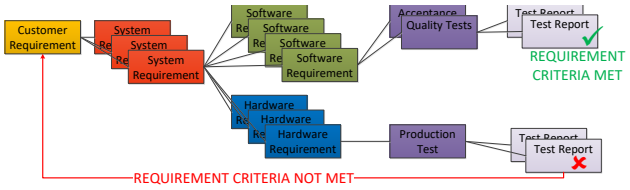

Fig. 2. Interaction between the objects: Requirements and Tests.

C. *Projects*

Requirements may be reused and reassigned to several projects and products. For instance, the same train-to-wayside data connection can be present in different products of different projects, although it may not be implemented with the same protocol. Requirements may not be interdependent.

D. *Traceability to documents*

The solution must offer a traceability mechanism from a requirement to any existing documentation. For instance, a System Requirement object in the JIRA database needs to be aware of all System Requirement Specifications into which it is exposed. Documents are in MS Word.

E. *Reporting*

The solution needs to generate requirement traceability reports. Therefore, the solution must tell if a Customer Requirement is fulfilled or how it is recognized as satisfied by a customer. This information needs to be generated into a report, which will then serve as a proof for traceability criteria and project milestone completion. The reporting allows to trace requirement statuses, such as those presented in TABLE I.

TABLE I. REQUIREMENT REPORTS EXAMPLES

Report	Description
Linked	% or # of requirements covered by a requirement upstream
Not linked	% or # of requirements uncovered upstream
Tested and verified	% or # of requirements covered downstream and tested (with a pass status)
Linked + resolved (tested or justified by a traceability comment)	% or # of requirements covered and tested (with a pass status) OR documented downstream
Not linked + resolved	% or # of requirements unallocated upstream but tested and verified

F. *Baseline*

The solution must offer to generate a snapshot of any given set of requirements at specified times, such as project phases, gates and milestones, and for project systems and sub-systems. Baselines need to offer search features, in order to analyze requirement changes that might have occurred between two snapshots.

IV. CONFIGURATION OF THE SOLUTION DEPLOYED

A. *JIRA*

We successfully configured JIRA in order to manage requirements the way described in the above section. We added the issue types, custom fields and link types for our requirements, tests and change requests. Evaluations were made, for instance, that of having a single requirement issue type with an associated custom field to describe the requirement sphere (customer, system, hardware, software) or to have separate issue types, one per sphere. Another configuration: do we use the built-in "component" field from JIRA to map sub-systems or do we create separated custom fields to manage them. The "label" option is also available.

The JIRA issue is configured to report such details, as shown below (Fig. 3).

Fig. 3. JIRA issue showing a System Requirement object with calculated Custom Fields.

In order to reuse existing requirements into several projects, we proposed to have JIRA projects which represent ISC "products." Products may contain different software and hardware implementation for a same system requirement (for instance, communication protocols may differ from one project to another). Traceability to document is kept by a textual custom field representing the document ID. This is suitable for multiple instantiation of a single requirement into several documents. We can address two-way automation either by parsing the documents update the JIRA database, or alternately use the ID field in JIRA to generate documents using third party tools.

B. *Third party add-ons*

One considerable challenge is to manage the traceability between requirements and to report it globally so that the information can be soundly used in order to follow system development progress. JIRA built-in reporting technology is limited unless users can code their own reports. Several third-party tools exist for facilitating report generation. Our evaluation led us to eazyBI. In order to allow validation and verification through multiple levels of requirements and tests, we explored third party plugins. We chose JIRA Misc Custom Field, as this option offers to extend JIRA with Calculated Custom Fields to let users define a custom field formula by using the JIRA SDK API in Java. We were therefore able to extract specific data from second-level requirements to generate custom traceability links which are sent to eazyBI for metrics generation. We show some of these fields in TABLE II. To complement, we also installed Impact for JIRA.

Luc Filion, et al., "Using Atlassian Tools for Efficient Requirements Management", in 2017 IEEE International Systems Conference (SysCon 2017) Proceedings, Pages: 446 – 451. © IEEE 2017 - DOI: 10.1109/SysCon.2017.

TABLE II. TRACEABILITY FIELDS GENERATED USING PLUGIN

Custom Field Name	Result (all are text string formats, issue=requirement)	Short description
Has Component or Sub-component	• issue have subcomponent(s) • issue have both component(s) and subcomponent(s) • issue have component(s)	Does the requirement has one or more allocated components and/or sub-components
Implements issues	Unique issue keys string (for instance "REQ-1, REQ-2, REQ-44")	The list of upstream requirements
Verified, Tested or Justified	• Some issues are not resolved • All issues are resolved • All issues are resolved with at least one justified • Justified • Satisfied	Determines the status of all the downstream requirements linked with the current requirement.

In this table, we assume that:

- Systems are made up of components (e.g. Controller) which can be composed of a hardware or software sub-components (e.g. Connectors, Firmware)
- Verified means that a requirement has all its derived and linked requirements in the state "Completed"
- Resolved means that a given requirement is linked with derived requirements which all have linked test reports with a "Pass" status
- Justified means that a given requirement is met by an external reference which is not a test (e.g. validated by design, data sheet, specification compliance, etc.)
- Satisfied is all of the above combined.

C. Configuration of eazyBI

eazyBI has been configured in order to report the custom fields presented above. eazyBI proposes to consolidate data measures through a 3-dimensional data model [14] and offers to access this data in a fast and independent database. Fig. 4 depicts this concept clearly.

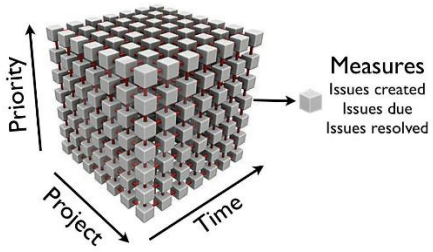

Fig. 4. eazyBI data model works with JIRA fields in several dimensions (for instance Priority, Project and Time).

We have access to JIRA basic fields as well as custom calculated fields that we programmed. These fields are imported in eazyBI as dimensions and measures are therefore calculated and displayed as charts and reports.

D. Confluence

Even though Confluence has been deployed, it is not targeted yet for writing document to be delivered to customers. Confluence does not offer enough features nor provide sufficient document quality to be used in our case. Confluence will be used for internal documentation only.

V. RESULTS

We analyzed several reports and charts from the eazyBI database in order to generate the metrics described in TABLE II. The calculated custom fields are represented as user defined measures we have access to.

1) Traceability Report

For instance, we can generate a traceability report (taken from Table I) showing the distribution of requirements for an on-going project Alpha.

	Customer Requirement	System Requirement	Software Requirement	Total
linked (parent issue)		183	2	185
not linked (no parent issue)	213	400	1	614
linked + resolved (tested)		8		8
linked + resolved (tested or justified)		1		1
not linked + resolved (tested or justified)	2	3	1	6
Total	215	595	4	814

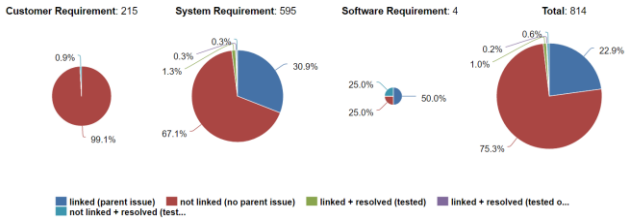

Fig. 5. eazyBI results in various formats, such as tables and pie charts

This report (see Fig. 5) allows us to quickly view the current requirement coverage level for our project based on the metrics we defined. This global report is a basis of work for us to analyze the requirements. We may drill into a specific report category for looking at targeted data. This is also useful to find irregularities in the requirement management data. eazyBI allows us to add or modify the report dimensions and measures by a drag and drop operation in order to refine the results obtained. For instance, we may confine the report above to a specific system component (for instance, a controller). Such a report may be used as an internal fragmented data to plan and prioritize the work to do on a specific system. We may also inverse the table above and sort the result by system component instead. Each dimension in eazyBI can have hierarchies that are used for measure aggregation. For instance, linked requirements are split into several components.

2) External Document References

ISC submits to its customers several documents describing the systems delivered (for instance, a Requirements

Luc Filion, et al., "Using Atlassian Tools for Efficient Requirements Management", in 2017 IEEE International Systems Conference (SysCon 2017) Proceedings, Pages: 446 – 451. © IEEE 2017 - DOI: 10.1109/SysCon.2017.

Specification) in MS Word format. Links between any requirement stored in the JIRA database and its references in various Word documents are always maintained. As mentioned earlier, we added a JIRA custom field to link requirements with external documents. This allows to create a documentation coverage report. For instance, Fig. 6 shows the coverage results for all the requirements in a given SyRS. This allows ISC to have an instantaneous status for any system described in any document, at any time. This information is extracted from the database is precious, as it allows to provide customers with a real-time status regarding a system progress.

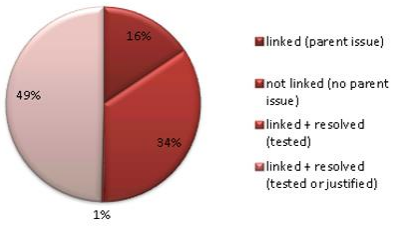

Fig. 6. System Requirements Specification (SyRS1000-0063-01) coverage

Another analysis example shows that 18% of the requirements allocated to hardware cannot be verified, because they are not associated to any components/subcomponents of the system. This report allows ISC to validate that the requirements respond to the system needs. Since any project may inherit existing requirements from other products or projects, inconsistencies may have occurred while importing them to the new project Alpha. Furthermore, this quick analysis allows to eliminate hardware requirements that may not be necessary to implement (because they are not linked with system or customer requirements).

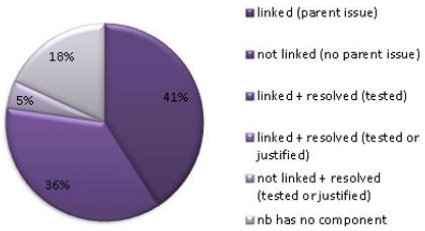

Fig. 7. Hardware Requirements coverage report related to components and subcomponents.

B. *Impact for JIRA*

Using the Impact for JIRA plugin allows to analyze the requirement hierarchy dynamically and access the JIRA database on the spot. This frontline tool can quickly show JIRA links between a set of requirements, so that they can be analyzed further in depth. For instance, we may extract and display all the software requirements from a specific module that are neither covered nor justified (e.g. orphans).

Also, we can extract the metrics obtained from eazyBI (for instance, the 8 requirements in Fig. 5 that are "linked and resolved") and input them into Impact. Fig. 8 shows an excerpt of this analysis in Impact. It has been exported in Excel format and can be used as a traceability report.

Traceability matrices which are sometimes used to demonstrate coverage are not necessary as this solution allows to obtain the same information but in a structured style instead.

Key 0	Key 1	Key 2	Key 3	Status	Assignee
REQ-69				To Do	Unassigned
is implemented by	REQ-575			To Do	Unassigned
	is implemented by	REQ-853		Closed	Luc Filion
REQ-70				To Do	Unassigned
is implemented by	REQ-482			To Do	Unassigned
is implemented by	REQ-233			To Do	Luc Filion
	is implemented by	REQ-853		Closed	Luc Filion
	is implemented by	REQ-863		Resolved	Unassigned
		is implemented by	REQ-864	To Do	Unassigned

Fig. 8. Impact Traceability Report excerpt

C. *Numbered Results, Expectations and Extrapolations*

At the time of writing this paper, the requirements management solution has been deployed and measures are being taken on a given ongoing customer project (Alpha). The first results are above expectations. We expect to significantly improve requirement management activities directly and indirectly, compared to a manual requirement management in Excel.

TABLE III. RESULTS AND BENEFITS

Benefits	Examples of measures, and Extrapolations
Direct	• 20% in time reduction for management of requirements, expecting better project control and better requirements coverage • 5-10% Reduction of "after-the-fact" errors
Indirect	• Better project value evaluation throughout gates • Time and Costs savings following reduction of errors • Time saved by having requirements more easily accessible to developers, integrators and testers (through JIRA) • Improved project quality

1) *Direct Benefits*

We have already calculated a 20% time reduction for managing the requirements, which mostly comes from a more frequent interactions with the requirements. Increased frequency is made possible by an easier accessibility to requirements. Having a more dynamic relation with the requirement database reduces the validation and verification effort. Therefore, we expect this to materialize in a 5% to 10% gain by causing less after-the-fact errors. The validation and verification data is made available by the JIRA database and through eazyBI and Impact reports. These also allows features implementation cost savings.

2) *Indirect Benefits*

ISC projects go through several gates. Therefore, it is expected that each gate be more precise in terms of project progress in regards to the customer and internal requirements coverage. The project cumulated value can be directly deducted and shared with the project stakeholders.

Luc Filion, et al., "Using Atlassian Tools for Efficient Requirements Management", in 2017 IEEE International Systems Conference (SysCon 2017) Proceedings, Pages: 446 – 451. © IEEE 2017 - DOI: 10.1109/SysCon.2017.

Besides, we believe that the solution induces indirect advantages by letting the software and hardware engineers have shared access to requirement objects. Because engineers already use JIRA to plan their development activities (tasks, stories, sprints, etc.), they have the possibility to consult the requirement database more routinely.

All these indirect benefits have a significant impact on the systems quality.

Future Works and Conclusion

Requirement Specifications and Test Plans are currently manually written. At the time of writing this paper, we are developing a solution for automating the document generation for all specifications, including changes history. Additionally, it is expected that we extend the list of available eazyBI reports for system analysis. We will enrich custom fields for managing the system components more effectively, and finally, we intend to improve the performance for the report and document generation.

Besides allowing ISC to achieve CMMI-2, these results lead to significant project cost and time reductions. This allows ISC to obtain a customized solution for their specific needs. By doing this work, we overcame the limitations of JIRA for requirements management (as stated in [4]), and we brought ISC, SME and the system engineering community a very efficient and low cost approach to requirements management with JIRA.

References

[1] Software Engineering Institute, CMMI for Development, Version 1.3, Pittsburgh, PA: Carnegie Mellon University, CMU/SEI 2010-TR-033, 2010.

[2] Pohl, K. Requirements Engineering: Fundamentals, Principles, and Techniques. 1st Edition. Springer Publishing Company, Incorporated ©2010

[3] Sud, Rajat R. and Arthur, James D. (2003) Requirements Management Tools: A Quantitative Assessment. Technical Report TR-03-10, Computer Science, Virginia Tech.

[4] Seilevel's 2016 Requirements Management Tools Evaluation Report. http://www.seilevel.com/cio-article-rm-tool-evaluation-report/

[5] JIRA Web Site. https://www.atlassian.com/software/jira

[6] Confluence Web Site. https://www.atlassian.com/software/confluence

[7] eazyBI Web Site https://marketplace.atlassian.com/plugins/com.eazybi.jira.plugins.eazybi-jira/server/overview

[8] Oppenheim B. Lean for Systems Engineering. Wiley 2011.

[9] Ries, Eric. The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. CROWD. 2011.

[10] Powel Douglas, B. Agile Systems Engineering. Morgan Kaufman. 2015

[11] eazyBI Reports and Charts https://docs.eazybi.com/display/EAZYBI

[12] JIRA Misc Custom Fields add-on https://marketplace.atlassian.com/plugins/com.innovalog.jmcf.jira-misc-custom-fields/server/overview

[13] Impact for JIRA https://marketplace.atlassian.com/plugins/ca.nuum.impact/server/overview

[14] eazyBI Data model. https://docs.eazybi.com/display/EAZYBIJIRA/Data+model

Luc Filion, et al., "Using Atlassian Tools for Efficient Requirements Management", in 2017 IEEE International Systems Conference (SysCon 2017) Proceedings, Pages: 446 – 451. © IEEE 2017 - DOI: 10.1109/SysCon.2017.