OpenReq Issue Link Map: A Tool to Visualize Issue Links in Jira

Clara Marie Lüders
The Qt Company
Espoo, Finland
clara.luders@qt.io

Mikko Raatikainen
University of Helsinki
Helsinki, Finland
mikko.raatikainen@helsinki.fi

Joaquim Motger
Polytechnic University of Catalonia
Barcelona, Spain
jmotger@essi.upc.edu

Walid Maalej
University of Hamburg
Hamburg, Germany
maalej@informatik.uni-hamburg.de

***Abstract*—Managing software projects and products gets more and more complicated with an increasing project and product size. To cope with this complexity, many organizations use issue tracking systems, where tasks, bugs, and requirements are stored as issues. Unfortunately, managing software projects might still remain chaotic even when using issue trackers. Particularly older projects with lots of issues and many links between the issues, make it hard to maintain an overview of the dependencies, especially when tens of new issues get reported every day. We present a Jira plug-in tool that supports developers, project managers, and product owners in creating and keeping order of issues. Our tool visualizes the issue links, helps to find missing or unknown links between issues, and detects inconsistencies.**

***Index Terms*—Requirement Dependencies, Data-Driven Requirements, Issue Tracking Systems, Recommendation Systems, Release management, Similarity/Duplicate Detection**

I. Introduction

Many software companies keep track of their work in issue tracking systems, such as Jira, Bugzilla, or Github. Stakeholders can describe and store requirements—called issues or tickets in these systems—in the form of epics, user stories, tasks, bugs, feature requests or other defined types. Issues have multiple attributes such as title, description, status, resolution, environment, priority, assignee, release version, reporter, or links to other issues. Depending on the issue tracking system and the users of the system, there can be a variety of different link types. Common types are parent-child, duplicates, dependencies, similarities, relations, and work breakdowns. In this work, we refer to dependencies between issues as *links* adhering to the terminology in Jira. Additionally, stakeholders can comment issues and track their life-cycle. We studied the challenges encountered in managing software projects and build a tool to support this activity. We are evaluating it with The Qt Company in their Jira.

Keeping an order, which means that all information regarding issues is stored and therefore visible in the system, is hard in large issue tracking systems. Especially in open-source projects where a community submits hundreds of tickets weekly. Organizing new issues and planning releases becomes a demanding, time-eating task with no end in sight—a frustrating situation for developers, project managers, and product owners. Links influence product management because the product manager or product owner needs to make sure that every issue linked for a feature is taken into account if this feature is supposed to be in a release. With many issues keeping track of relevant issues and links gets complex, e.g. Qt's public Jira contains (May 2019) over 111,959 issues, out of which 27,462 issues have at least one link to another issue, with a total of 24,857 explicated links. Another difficulty during release planning is that links can be known but the person with the knowledge does not have the rights to add the link, they can be wrong, forgotten, or are simply not known. A simple example is that a bug-type issue is reported by two different persons, but as they do not know it, the issues are not linked as duplicates.

Organizing and understanding this spiderweb of issues needs to be facilitated One solution is to visualize this with a graph, but issue tracking systems do not have built-in support to visualize the links of issues. Users only see a list of all direct links of a selected issue and not further beyond. E.g., in the view issue page of QTBUG-55604 in Qt's Jira, only three issues are visible to users, but in fact there are a total of 19 issues linked transitively to the issue. Such a network of links can become huge and complex. E.g., there is a link network of size 6755 in which the longest shortest distance is 51 links across multiple Jira projects. Grasping the dependencies and issues without any visualization is not feasible.

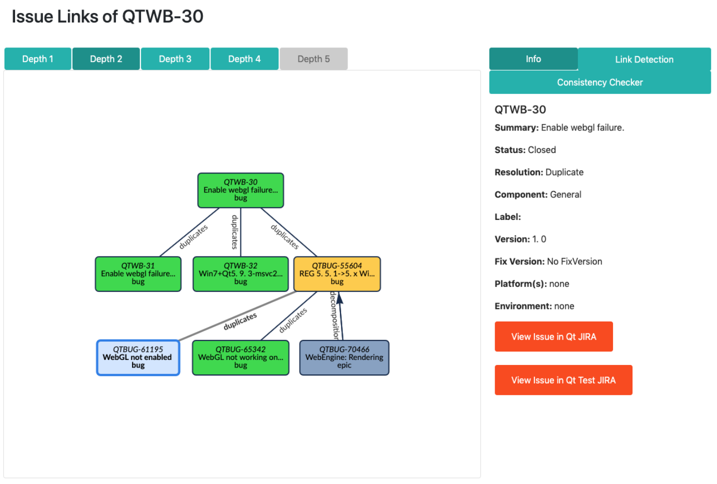
The European Horizon 2020 project OpenReq (https://openreq.eu/) aims at solving this problem. One goal is to develop an open-source Jira plug-in that any company can use to help manage their projects. Besides visualizing the link map of issues (see Figure 1), the tool uses recommendation systems to enable users to find missing or unknown links. Additionally, users can check the consistency of an issue network with regard to the release plan(s) [1].

II. The tool: OpenReq Issue Link Map

A. Overall Architecture

The OpenReq Issue Link Map is currently a service-based tool that visualizes the link map of issues in Qt's Jira with the goal to have it as a Jira plug-in. So far, in order to get a better understanding of the whole picture , users have to explore the links one after another in Jira. With the Issue Link Map visualization users can see all linked issues at a glance,.The front-end is a web-based interface while the back-end[1] consists

[1]A detailed description was submitted to the industry track of RE'19


Issue Links of QTWB-30
Depth 1
Depth 2
Depth 3
Depth 4
Depth 5
QTWB-30 Enable webgl failure... bug
duplicates
duplicates
duplicates
QTWB-31 Enable webgl failure... bug
QTWB-32 Win7+Qt5. 9. 3-msvc2... bug
QTBUG-55604 REG 5. 5. 1->5. x Wi... bug
duplicates
duplicates
decomposition
QTBUG-61195 WebGL not enabled bug
QTBUG-65342 WebGL not working on... bug
QTBUG-70466 WebEngine: Rendering epic
Info
Link Detection
Consistency Checker
View Issue in Qt JIRA
View Issue in Qt Test JIRA

QTWB-30

Summary: Enable webgl failure.

Status: Closed

Resolution: Duplicate

Component: General

Label:

Version: 1. 0

Fix Version: No FixVersion

Platform(s): none

Environment: none

Fig. 1: Issue Map with depth 2 of QTBUG-30 in OpenReq Issue Link Map

of several OpenReq microservices that work together in a choreographic manner.

B. Background Services

1) Graph of Links: Based on issues and their links, a graph of links facilitates the visualization of an issue link map. The graph is maintained as a separate process to allow for a real-time response of a link map. This is necessary in the case of Qt's Jira since the number of issues is too large for a synchronous issue retrieval and analysis in Jira directly.

2) Link Detection: Additional links are detected from the text. In order to find the duplicates, all issues of a single project are taken and their title and description are compared against each other to decide if they have similar content. Moreover, a cross-reference detection checks the comments of an issue for the mention of another issue, which can indicate a link. This happens due to the open-source nature of Qt: Everybody can create issues, but not everybody is able to create links between issues. In order to work around this, users often comment if they think a link exists. A maintainer then has to add the link manually.

3) Consistency Checker: The consistency checker [2] verifies that the release plan of an issue link map is consistent by applying constraint solving technologies: First, all child issues, which have the same or higher priority, must not be assigned to a later release. Second, any required issue must not have a later release or lower priority. Third, all links from a duplicated issue are inherited by the duplicate issue.

C. User Interface

The user interface consists of two main parts, the visualization and the features explained above (see figures in Appendix). As seen in 1 the link network is shown as a graph on the left side where a user can select the depth of the map, Depth means the distance from the selected issue. The link map can be navigated by clicking any issue in the map. On the right hand side, users can see general information about the issue from Jira. Above the information box, users can switch to two features: link detection and consistency checker. When the link detection is called, users are presented with a list of the 5 recommended links that can be accepted or rejected. If they accept the link, they must select its type as well. When the consistency checker is clicked users see if the issues contained in the issue link map and the corresponding released are consistent. The different releases are also shown underneath the result.

III. Future Work

The tool is still in development and thus the capabilities of the link detection can be improved upon, e.g. the choice of the users to accept and reject links can be used to train and refine the system, as a human-in-the-loop approach. Additionally, it is planned to visualize what part makes a certain issue link map inconsistent with the option to automatically repair the inconsistency. The Qt trial started in January of 2018 and the tool is being developed and evaluated in this trial, the OpenReq project is scheduled until the end of 2019. We use some AI technologies in this trial to support open-source issue tracking systems and make them more manageable. In the long run, we plan to analyze user acceptance of AI technologies that can aid their work. A problem here is that AI is usually a ”black box”. By integrating users in the decision process and showing them the reasoning of the system we hope to increase AI acceptance.

Acknowledgement

The work presented in this paper has been conducted within the scope of the Horizon 2020 project OpenReq, which is supported by the European Union under the Grant Nr. 732463

References

[1] A. Felfernig, M. Stettinger, A. Falkner, M. Atas, X. Franch, and C. Palomares. Openreq: Recommender systems in requirements engineering. In A. Felfernig, M. Stettinger, A. Falkner, M. Atas, X. Franch, and C. Palomares, editors, *RS-BDA'17*, pages 1–4, Graz, Austria, 2017.

[2] M. Raatikainen, J. Tiihonen, T. Männistö, A. Felfernig, M. Stettinger, and R. Samer. Using a feature model configurator for release planning. In *Systems and Software Product Line Conference - Vol. 2*, 2018.

Annex

A. *How the tool will be presented*

We will present *OpenReq Issue Link Map* in an interactive and live tool demonstration and with a poster explaining the problem and showcasing the features of the UI.

Interactive Demonstration. The tool can be reached under the following url: https://api.openreq.eu/openreq-issue-link-map/. At the conference, interested persons are free to interact with the tool as they wish. Some features, such as accepting recommended links, will not be possible since the users do not have the sufficient access rights for it.

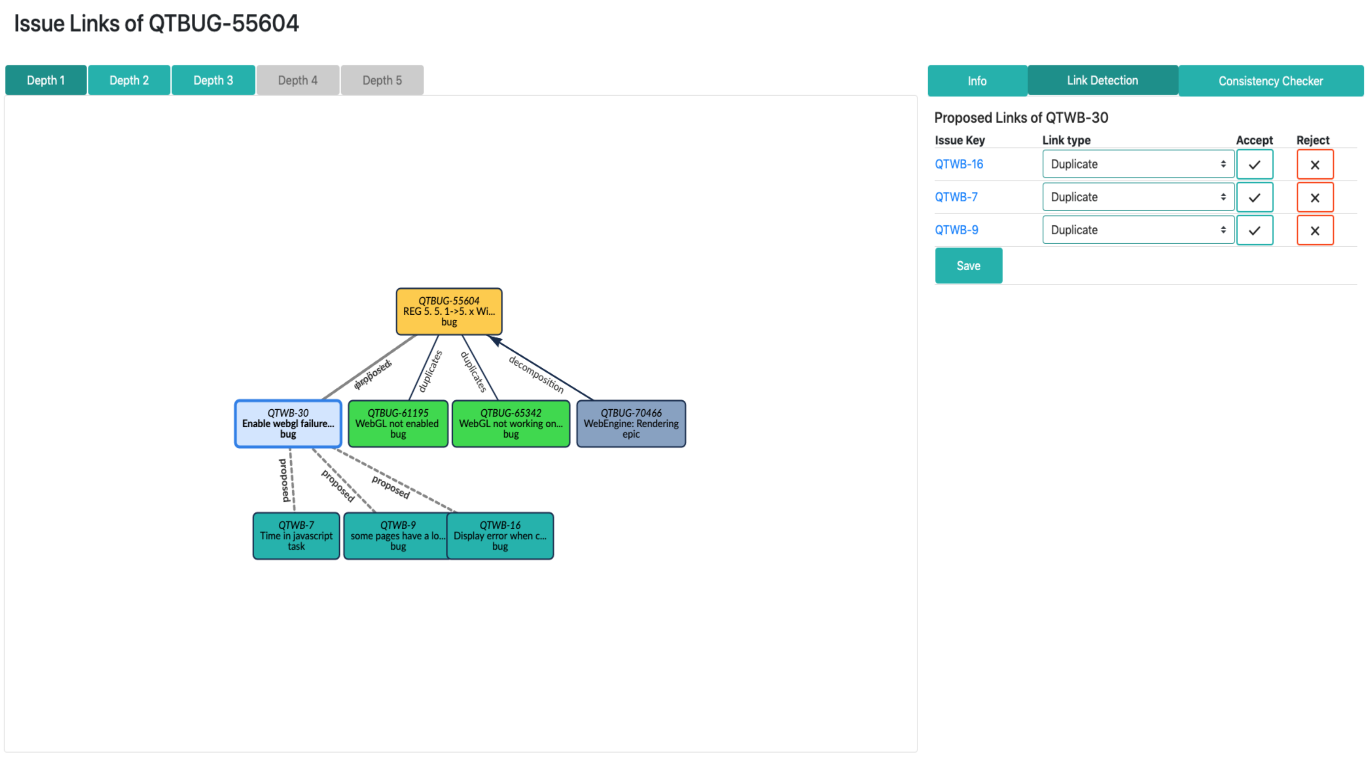
The European Horizon 2020 project OpenReq and its goals will be introduced as well. We further explain how and why we are developing the tool and in which industry scenarios it is currently tested and useful. Additionally, we will describe how the tool can be used in other circumstances apart from Qt's environment.

Poster. A poster that visualizes the problem and the features will be provided, so that interested persons can learn about the tool and what it does prior to trying it out. This way the audience has a way to familiarize with the tool and a guide to refer to while they are using the tool. As the UI is not described in full detail in the tool paper itself, the focus of the poster will be on the different UI elements and their functions so that the user can achieve a better understanding of the OpenReq Issue Link Map.

B. *Further documentation*

Every component of the European Horizon 2020 project OpenReq is open source, and so are all microservices of *OpenReq Issue Link Map*[2]. Despite the code being publicly available, we also document all API endpoints of our components[3]. In our demonstration, we will direct the audience to both points since all parts of the OpenReq project can be used stand-alone in a requirements engineering context.

[2]https://github.com/OpenReqEU
[3]https://api.openreq.eu

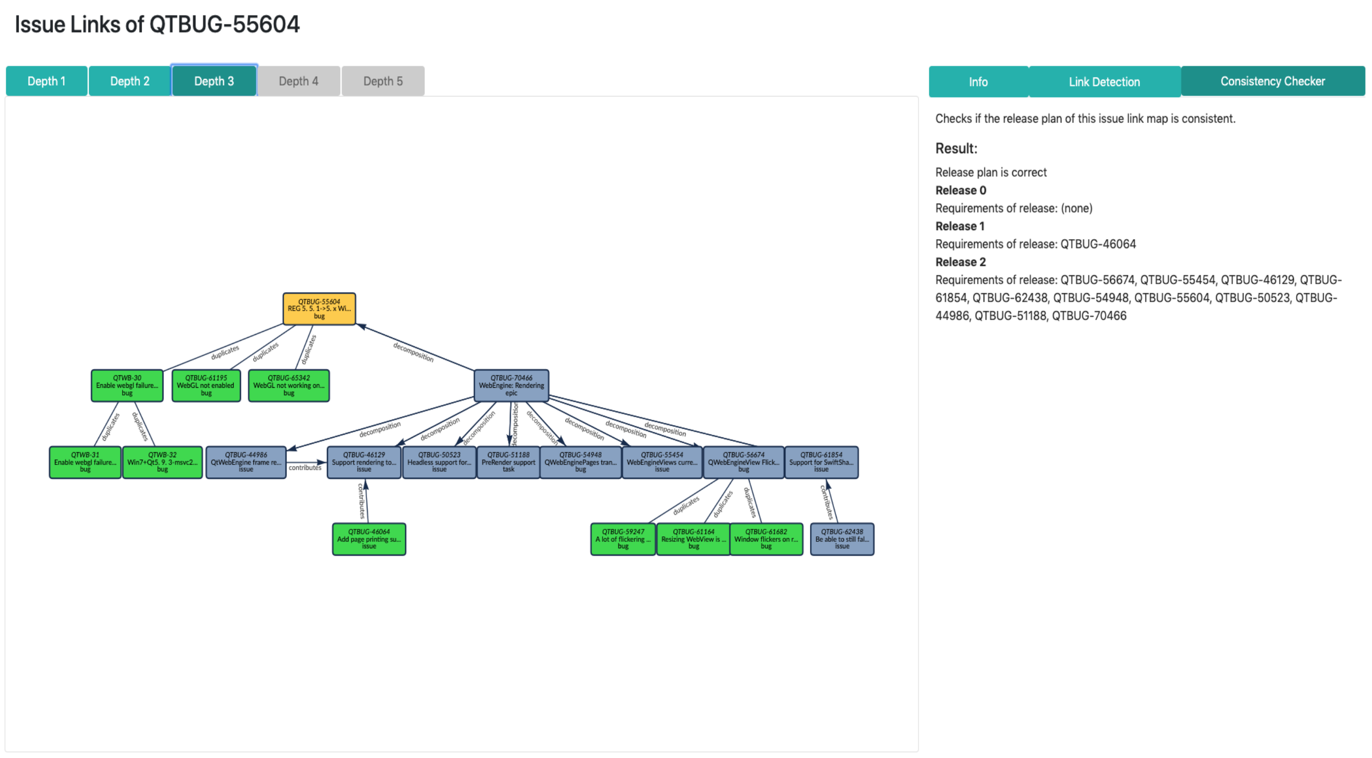


Issue Links of QTBUG-55604

Proposed Links of QTWB-30

Issue Key	Link type	Accept	Reject
QTWB-16	Duplicate	✓	✕
QTWB-7	Duplicate	✓	✕
QTWB-9	Duplicate	✓	✕

Fig. 2: Link Detection of Issue QTWB-30



Issue Links of QTBUG-55604

Checks if the release plan of this issue link map is consistent.

Result:

Release plan is correct

Release 0

Requirements of release: (none)

Release 1

Requirements of release: QTBUG-46064

Release 2

Requirements of release: QTBUG-56674, QTBUG-55454, QTBUG-46129, QTBUG-61854, QTBUG-62438, QTBUG-54948, QTBUG-55604, QTBUG-50523, QTBUG-44986, QTBUG-51188, QTBUG-70466

Fig. 3: Consistency Check of Issue Link Map of Issue QTBUG-55604