Let’s try the Linux way of working

Anforderungsmanagement

Let’s try the Linux way of working

Wer sind wir?

Vehicle base software & middleware
development tools
cloud-based operations services
cybersecurity solutions
end-to-end engineering

Philip Partsch
philip.partsch@etas.com

Lead Systems Requirements Engineer
Process Responsible

Let’s try the Linux way of working

Motivation

Was Sie umtreibt


Kooperationen


Verlinkung von Artefakten


Automatisierte Qualitätsebenen


Reporting

Was Sphinx-Needs zusätzlich bietet


Open Source Software


Interface zu externen Tools / Service


Community


Professioneller Support

Let’s try the Linux way of working

Von Docs-As-Code zu X-As-Code

Docs-As-Code

– Wir nutzen zur Pflege der Dokumentation die gleichen Methoden und Tools wie für Code

– Semi-automatische Erstellung der Dokumentation aus Quellcode

Beispiele sind Doxy-Gen oder Sphinx

X-As-Code

– Wir pflegen nicht nur die Dokumentation, sondern auch andere Entwicklungsartefakte wie Code

– Einfaches Verlinken von Informationen

Beispiele für Informationen:
– Anforderungen
– Analysen
– Architektur
– Test Spezifikationen
– Test Ergebnisse

Let’s try the Linux way of working

Feedback

Wie viele Tools nutzen Sie für Ihre Entwicklung?

Wie lösen Sie die Problematik der Verlinkung?

Let’s try the Linux way of working

Was ist Sphinx & Sphinx Needs?

– **Was ist Sphinx?**

Sphinx ist ein Tool, das schnell und einfach eine schöne Dokumentation erzeugt. Es bietet Unterstützung bei der Dokumentation von Quellcode. Sphinx nutzt reStructuredText als markup language.

– **Was ist ein Need?**

Ein Need ist ein Klassen ähnliches Objekt, das aus einer ID, einer Beschreibung sowie Eigenschaften und Links zu anderen Needs besteht.
Es kann exportiert, importiert, gefiltert und damit visualisiert werden.

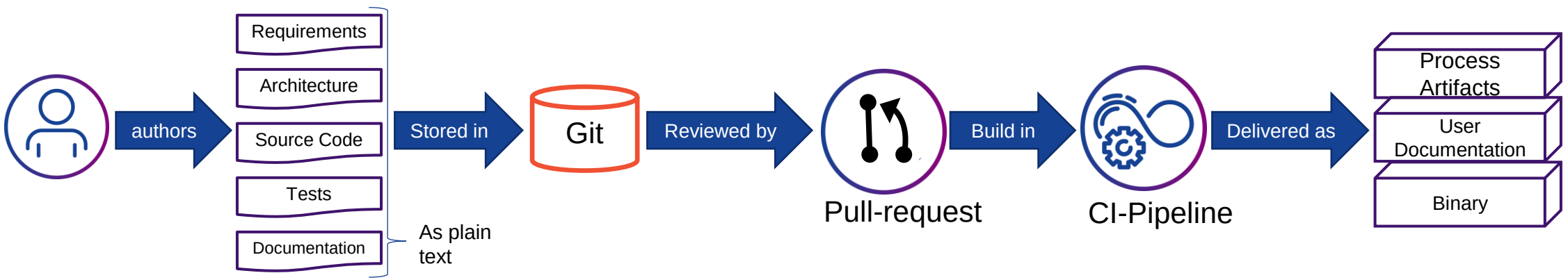
– **Was ist Sphinx-Needs?**

Sphinx-Needs ist eine Erweiterung von Sphinx.
Die Kern-Funktionalität ist das Verlinken und somit die „Traceability“ von Needs in Spezifikationen.

Beispiele sind Entwicklungsprozesse selbst oder Entwicklungsartefakte

Let’s try the Linux way of working

Wie wird es genutzt?



Let’s try the Linux way of working

Das war zu abstrakt: Wie nutze ich es konkret?

Let’s try the Linux way of working

Kann ich eine Beispiel sehen?

```
.. sw_req:: Merge python dictionaries
   :id: SWRQ_MERGE_DICTS
   :status: verified
   :satisfies: CSTRQ_MERGE_DICTS

   The python module `merge_dicts` shall provide
   a function to merge a list of
   python dictionaries to one dictionary.
```

Software Requirement: **Merge python dictionaries**

SWRQ_MERGE_DICTS

status: verified

generated by: EVAL_STAKE_MANY_DICTS

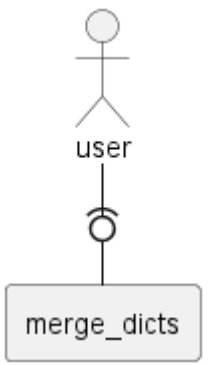
satisfies: CSTRQ_MERGE_DICTS

tested by: TS_MERGE_DICTS

The python module *merge_dicts* shall provide a function to merge a list of python dictionaries to one dictionary.

```
.. needuml:: UML Diagram

   Actor user
   {{uml('M_MERGE_DICTS')}}
   user -(0- M_MERGE_DICTS
```



Let’s try the Linux way of working

Live Demo

Demo mittels

https://github.com/PhilipPartsch/ReConf2023-Product-As-Code

Let’s try the Linux way of working

Quellen & Links

Framework und Dokumentationen

- Docutils
- sphinx-doc.org
- sphinx-needs.com
- Documentation without Frustration
- Pandoc

Sphinx Erweiterungen

- sphinx-test-reports.readthedocs.io
- sphinx-collections.readthedocs.io
- sphinx.ext.autodoc
- sphinx.ext.inheritance

Thank you!