ABSTRACT
With the industrial success of knowledge-based systems new requirements with respect to Knowledge Engineering processes arise. Besides advanced knowledge acquisition tools, novel techniques for the quality assurance need to be established in order to maintain a safe development process. In Software Engineering, the application of continuous integration as a collection of practices has proved to be suitable for this task. In this paper, we transfer the general ideas of continuous integration from Software Engineering to Knowledge Engineering, and we demonstrate the implementation of a continuous integration tool into a state-of-the-art Knowledge Engineering workbench.

Categories and Subject Descriptors
K.6.3 [Software Management]: Software maintenance; I.2.1 [Applications and Expert Systems]: Industrial automation

General Terms
Knowledge Engineering, Software Engineering, Agility

Keywords
Continuous Integration, Testing, Knowledge Engineering

1. INTRODUCTION
During the last decades, the construction of knowledge-based systems has grown from an academic discipline into an industrial development process. In the past, traditional development processes and tools have been developed [1], while more recently agile developments techniques [2] have been focused. With the industrial impact of such systems, new requirements are posed to Knowledge Engineering techniques with respect to the development and maintenance process of knowledge bases.
In this paper, we argue that the ideas and practices of continuous integration—as known from Software Engineering—can be transferred into Knowledge Engineering practices, and we motivate that continuous integration also significantly improves the development process in Knowledge Engineering. We demonstrate its application in combination with the knowledge modeling tool KnowWE, a Semantic Wiki empowered by a plugin to allow for continuous integration. From this prototype implementation, it is easy to see that continuous integration can be implemented in most Knowledge Engineering tools available today.

2. CONTINUOUS INTEGRATION IN KNOWLEDGE ENGINEERING

In this section, we propose the use of continuous integration practices for Knowledge Engineering, and we motivate that the development of knowledge bases can also benefit from (tailored) continuous integration.

Knowledge Base Repository. A central repository stores the artifacts of the knowledge base together with older versions. A repository should not only store the formalized parts of the knowledge base but also includes less formal parts relevant for the development process, such as organizational documents, sheets etc. A central knowledge base repository provides the essential infrastructure of a (distributed) development project, allowing all knowledge engineers to contribute to the project.

Automated Building of the Knowledge Base. A build is an executable artefact of the knowledge base, that is ready for deployment into the productive setting. The developed knowledge base needs to be transferred into such a “binary version”, that is later used to be deployed into practical use. A CI system should be aware that the knowledge base can be automatically transferred into an executable format, which can be accessed by the users.

Automated Tests of Current Builds. When introducing CI for knowledge bases, it is essential to provide automated tests for the knowledge. A test is automated, if the expected & correct result of the test is known apriori and it can be applied without manual interaction. After its execution the computed results are compared with expected results automatically. If they are not consistent, a development flaw has been detected and can be communicated to the developers. Such a test suite does not require human interaction, but simply returns failure or success when the tests have been executed. In the literature, there exists a comprehensive collection of validation and verification methods for different knowledge representations, see for instance research on validation [6] and verification [7]. By means of automated tests the knowledge base becomes “self-testable”. In consequence, tests can be inexpensively applied after every change. Frequent testing with a sufficient test suite increases the confidence of the domain specialists and the engineers involved [8].

Frequent Integration of Knowledge. It should be easy to integrate new developments of the knowledge base into the deployed production systems, for instance, extensions of the terminology or new rules. That way, the changes of the knowledge base can be integrated frequently and thus immediate feedback of the quality and utility can be generated. A necessary prerequisite of frequent integration is the existence of automated tests and version control. The first one is required for generating quality feedback, the latter one is needed to fall-back to an older version when problems with the latest integration of new developments occur. Frequent integration is a fundamental process, when peer developers are working in a distributed environment; then the (positive and negative) effects of the particular changes are propagated timely with frequent integration.

Easy Access and Deployment. Access to builds, i.e., executable versions of the knowledge base including the corresponding test results, are important in continuous integration. For a development tool supporting continuous integration, it should be simple to download the latest build, but also previous build versions. These builds should be directly usable in a productional environment.

The easy access of build versions simplifies the flexible adaptation of the knowledge base and thus improves the agility of the development project. The possibility to easily back-track to older versions of the knowledge base makes potentially wicked development steps less risky and therefore more attractive for the contributors.

The CI Dashboard. In the previous paragraphs, we discussed fundamental practices and techniques for continuous integration. For its practical use, the results of the development process and its continuous integration need to be communicated to the knowledge engineers. In Software Engineering, tailored dashboards have been proposed as an effective visualization technique to track the development process. At any time, a dashboard shows the overall quality state of the current version of the entire project at one glance and supports further acting if necessary:

- The creation of a new build can be started from the dashboard.
- It generates and displays a summarization of generated feedback of the latest build, e.g., the results of the test suite. Links to detailed reports are given.
- It provides access to previous builds and the corresponding changes.
- A weather report shows the healthiness of the knowledge base considering the past period of development, i.e., an assessment of the latest builds with respect to their quality.

A CI dashboard makes effects of the current development activities visible to everyone involved in the development process, i.e., knowledge engineers, domain specialists, or end-users.

In summary, the use of continuous integration provides evident benefits for the development of knowledge bases:

- Reduced risk during the development and evolution of a knowledge base: Thanks to frequent integration and automated tests, it becomes easier to detect problems early, for instance, when the latest changes have created anomalies. The reduction of risks allows for an agile process model, where changes can be flexibly
integrated during the development. In case of problems, the frequent integration and tests will uncover problems quickly and thus will make debugging very easy.

- With continuous integration it is possible to provide a running system at any time of the development process. With every integration, a productional build is created and archived by a version control system.

That way, independently of the current development stream, always a reliable version of the system can be downloaded and used, when necessary.

In the following section, we propose a Semantic Wiki as a Knowledge Engineering tool integrating CI practices quite intuitively.

3. CI WITH KNOWWE

Semantic Wikis [9, 10] are an extension to the concept of usual wikis [11], where besides informal content (e.g., text, figures) also formalized relations of domain knowledge are captured. These formalized relations are inserted into the wiki by explicit annotation methods, for instance certain markup or input forms; the made annotations are forming an (executable) knowledge base. That way, a Semantic Wiki serves as a collaborative Knowledge Engineering tool. The well-known wiki interface provides a large degree of freedom in structuring, and it allows for a workflow of incremental formalization.

![Figure 1: Continuous integration in the Knowledge Engineering tool KnowWE.](image)

In the following, we present the continuous integration support of the Semantic Wiki KnowWE [12]. Figure 1 sketches the workflow of continuous integration within the Knowledge Engineering tool KnowWE. On the left, one can see the contributors editing the knowledge using a standard web browser. The wiki server is provided with version control and a CI plugin, that allows for the continuous integration of knowledge bases. The initial continuous integration tool was published as a KnowWE plugin in August 2010 and refined/extended in the following months. It is contained in the default installation and was already used in a number of academic and industrial applications in technical and medical domains.

The tool can be configured easily to support tailored quality management for the respective project. Registered automated tests are performed on new states of the wiki knowledge base and give verbose feedback to the knowledge engineers by an integrated dashboard. The tests can be con-

---

![Figure 3: The trend of the knowledge quality considering the recent development is visualized by different weather icons.](image)
base. There, (sequential) test cases are defined within the wiki and attached to the continuous integration workflow by specifying a trigger mode.

4. CONCLUSIONS
We introduced the application of continuous integration practices in Knowledge Engineering and we argued that it can significantly improve the Knowledge Engineering process. Continuous integration is an effective set of practices to support agile development processes. While agile methodologies already play a major role in Software Engineering, they have also been a focus of research in Knowledge Engineering. Thus, we forecast that continuous integration also will emerge to a standard technique in practical Knowledge Engineering. In principle, every Knowledge Engineering tool is able to include support for continuous integration, when the following pre-conditions are fulfilled:

1. Support for automated tests: The tool needs to be able execute automated tests, so that the effects of knowledge modifications can be tracked.
2. Version control: Changes on the knowledge base need to be managed by a version control system, so that undesired effects can be backtracked easily.
3. A dashboard: The current state of the knowledge base need to be visualized and access to older versions of the knowledge base is necessary.
4. Easy deployment of production system: The tool needs to guarantee that a stable (potentially older) version of the knowledge base in a ready-to-deploy format is available at any time.
5. (optional) Distributed access to the knowledge base: The development effectiveness can be improved, when the tool supports a distributed access to the knowledge base.

Today, many state-of-the-art tools already support most of these pre-conditions.

5. REFERENCES