zAgile

Wikidsmart for ATLASSIAN CONFLUENCE®

Wikidsmart provides structure for your wiki, enabling:
- **Smart search to easily find precise information**
- **Automated maintenance of content and links**
- **Consistent data entry**
- **Deep interoperability with other applications and tools**

Guarantee Enterprise-Wide Adoption of your Wiki
The intrinsic nature of the wiki poses some challenges as its content grows. Wikidsmart enables you to overcome those challenges and guarantee widespread adoption of the wiki throughout your organization.

- *Your Expert "Wiki Gardener"*
  If a wiki is not constantly and accurately updated, its reliability is questioned within the organization. "Wiki gardening" describes those tasks necessary to keep information updated and accurate within the wiki.
- *Ensure consistent entry of content*
  Although it's easy to enter content, many times, particularly for repetitive data entry tasks such as writing requirements or specifications, data entry consistency is necessary each time a user enters the same type of data. Wikidsmart provides a choice of customizable templates to create forms so that data may be captured consistently. In addition, the content is captured semantically, so that the data's underlying meaning is captured. The semantic capture of data means that the data may be more easily found in the future or even combined with other information to infer additional content.
- *Automate generation and maintenance of content and links*
  Even worse than not updating content and links on a timely basis is updating the content erroneously. Wikis lack the mechanisms to automatically update information. Links must be copied and pasted by hand. With Wikidsmart, links and content itself can be automatically generated with embedded queries within a wiki page, to ensure real-time update of content.

Your Organization's Knowledge Repository

- *Easily find precise information*
  As the number of pages increases, the wiki's standard search method becomes increasingly inefficient. zAgile Wikidsmart's smart search enables users to find precise content easily via a search for the type of content (e.g., a document) and category of content (e.g., specification). Additionally, users can navigate through content contextually, having continual access to the tangentially related content.

Wikidsmart's semantic wiki engine turns your enterprise wiki into an information dashboard:

» Find precise information easily, across the wiki as well as other applications and tools

» Generate and maintain new pages with embedded queries, pulling content from the wiki and other applications and tools

» Achieve consistency of content across the wiki and other tools and applications

» Easily capture semantic annotations of existing content

» Create new content using semantic templates and forms, and populate other systems if desired

"zAgile seemed too good to be true, but they proved it to me. Before zAgile, we had the right tools, but we saw opportunities to gain more efficiency. … now we are even more productive across engineering, professional services, and product management teams."

**Niall Murphy**
VP of Engineering, Market6

Easy ALM and Productivity Applications All Within Your Wiki

- **For Teams with Existing ALM Tools**
  Your engineering team may wish to expose content from the ALM (Application Life-cycle Management) tools inside the wiki and also achieve deep interoperability between the wiki and the ALM tools. Wikidsmart provides a seamless way to integrate those tools into your existing processes, due to its infrastructure which is independent of the wiki.

- **Easy ALM, All Within Your Wiki**
  For any engineering team, a key capability is the tools to help with managing Project, Requirements, and Test Cases. Better yet, the tools should be integrated with a centralized repository. With Wikidsmart, those capabilities are built-in to the wiki, thanks to the underlying software engineering ontologies that drive Wikidsmart. Forms are available for each area of Project, Requirements, and Test Cases, and each may be customized to fit the desired processes of the team.
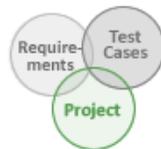
Easy, Centralized ALM in your Wiki

Your Wiki Becomes Your Enterprise Information Portal

- **A single coherent view of your enterprise information**
  Your organization has many applications and tools, but you have no single, coherent view of all information. With Wikidsmart, you have one coherent view of all information from all applications and tools.

- **Integrate tools and applications for deep interoperability**
  zAgile's infrastructure, which is independent of the wiki, integrates any tool or application. Therefore, any engineering tool or application can be semantically integrated with each other in a deep, contextual way.

- **Extreme traceability of information across organization**
  Due to Wikidsmart's ability to semantically integrate all information across the organizations teams, tools, applications, and processes, for the first time, you have the ability to achieve "extreme traceability."

zAgile Wikidsmart: *Just plug in your tools and applications for a comprehensive information collaboration dashboard across your teams.*

zAgile Wikidsmart provides semantic enablement of wikis and leverages the following high-level functional components:

» zAgile's Semantic Repository consisting of a set of ontologies and metamodels specific to a domain of interest. Ontologies are for Software Engineering are included, and others are available through zAgile professional services or other ontology providers.

» zAgile's Semantic Interface Layer provides wikis with access to the Semantic Repository via connectors

» zAgile's Semantic Plugin for Confluence provides the interface between Confluence and the zAgile Semantic Layer and supports macros for creating templates and forms in Confluence for annotating wiki pages.

**Challenges of the Traditional Enterprise Wiki**

Whereas a wiki provide an easy mechanism for users to collaborate, develop and maintain documentation and other content within an enterprise, its inherent wiki characteristics also come with some natural limitations:

- **Consistency of content:** First there is no ability of teams to collaborate by creating content in a consistent manner. Whether it is information related to projects, processes, people or some other domain-specific information, how consistently is that information captured? How easy is it to create content in a consistent manner across teams?

- **Search precision:** Search is limited to string matches within the content and can best return non-specific result sets. As the number of pages increases, this search method becomes increasingly inefficient.

- **Cross-referencing across pages:** Maintaining cross-references of information across pages is typically a tedious process that involves manually embedding links using wiki markup. These links, if not updated, can easily become stale, rendering the content unreliable and outdated

- **Integration of external information:** Information Integration with other applications and vice versa is mostly limited to data sharing via RSS feeds. If you are capturing information about product requirements in a wiki, then you may also want to integrate them with corresponding test cases, tasks, check-ins, etc. And you may want to do it both ways, i.e., integrate information from other tools into wiki pages but also pull some information from the wiki into other tools or applications. However, aside from page, space and section-level URLs, there is no other mechanism for such integration. Furthermore, the lack of integration with other applications and the lack of attribute level support limit the wiki to a static knowledge repository. If specific information changes on one page, the change is not automatically reflected everywhere else it may also appear. It must be manually updated, and if not, the content quickly loses integrity.

- **Context associated with pages:** The content categorization may be implied in the page hierarchy but there is no inherent structure to support it. Does a page depict a requirement, a process, a profile of a team member, some information about business partners, or instructions for a holiday party? There are limited mechanisms to define it through the use of labels but it is manual, arbitrary, and not structured. A page's content reflects some information or knowledge - but what is its nature? What is it describing? What are its relationships with other information concepts? You may be able to glean that off of the page title, its place in the page hierarchy, labels or its Space container but the page itself does not contain any attribute-level information.

- **Content organization is limited to mostly page-level hierarchies** This limitation is quite analogous to binders on a shelf. Designing and maintaining the page hierarchies, cross-referencing pages across spaces, tracking and updating them with the most current information and maintaining cross-reference links all require significant manual work. In the
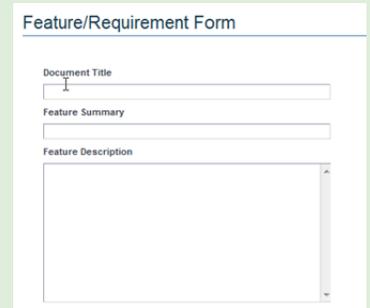
Wikidsmart enables you to create semantically enabled forms within your wiki
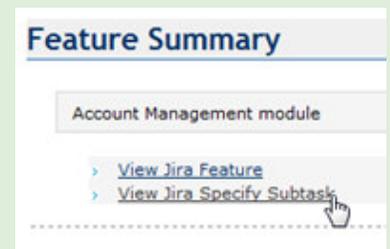

Content may automatically linked with external systems. This example shows links automatically connected to Jira

absence of this level of meticulous support, the wiki content quickly becomes outdated and unreliable. Within the context of Atlassian's Confluence enterprise wiki, the most typical organization method is through the use of Spaces which may imply or mimic some topic.

- **No wiki page attributes or metadata.** There is not an easy mechanism to store page-level attributes or properties within the context of the page itself so you cannot easily find a page written by a specific author, for a particular project, depicting a specific process or containing a specific section. This limits the level of granularity of the information or knowledge that is being captured and accessed within the page content.

Limitations of Traditional Enterprise Wiki Content
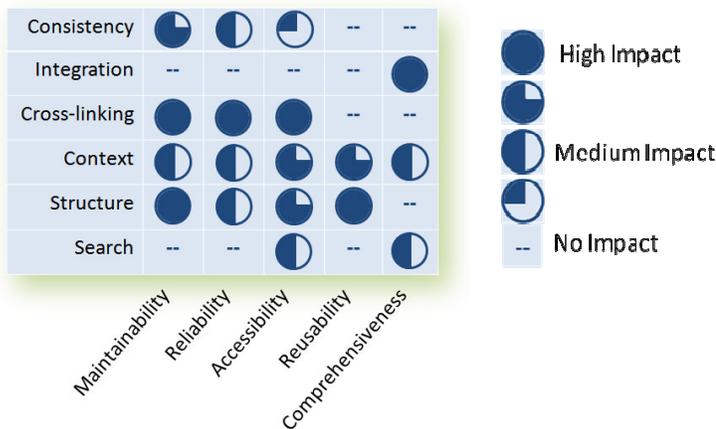
zAgile Wikidsmart's Semantic Enablement of Wikis
zAgile Wikidsmart's approach to semantic enablement of wikis addresses the natural limitations of wikis outlined earlier, and it also takes an architectural approach that overcomes the limitations of other implementations that strive to provide similar capabilities. The architectural benefits extend the ability of Wikidsmart to become an enterprise information portal, or "information collaboration dashboard."

zAgile's architectural approach:
- Separates the semantic repository from the wiki content so that the wiki functions not as the central and sole repository for both unstructured content and semantic data but as one of many applications that contributes semantically relevant data to it. This also allows users to have distributed and/or federated semantic databases.
- Facilitates two-way integration between wiki-based content and external applications that are also creators and consumers of related information. That is possible because the semantic repository is accessible to all applications, including the wiki, using the same interfaces. This level of semantic integration between the wiki and other applications provides a richer 'knowledge repository' that is capable of unifying an environment of disparate and heterogeneous applications and processes.

These extensions can be developed for any commercial or open source wiki, provided that the wiki supports a means for development of user extensions.

World Headquarters

zAgile, Inc.
101 California Street, Suite 2450
San Francisco, CA 94111
+1 (415) 946-8910 (Office)
+1 (415) 358-5924 (Fax)
www.zagile.com
info@zagile.com