**ViBRANT - Objectives**

biodiversity

ViBRANT is about connecting the people, data and science of the entire community.

powerful exemplar platform, the Scratchpads, for the existing gap between tools and services, providing a way to access digitized information should be expected in a near future, and the deep social and scientific impact of web 2.0 stimulates sharing of digital data and proposals of cybertaxonomy projects to study global biodiversity and climate changes. In the current context of the specimen studied in the field cannot provide the required information) are a good alternative.

Our contribution to ViBRANT is the implementation of an identification key generation web service, because we also have a significant experience in developing key generation tools such as MyKey (Gérard and Vignes-Lebbe 2010). The most common identification keys (described in Hagedorn et al. 2010) are the printed single-access keys published in almost all taxonomic revisionary work. But these printed keys may be difficult to use while collecting samples in the field or when consulting incomplete specimens. Hence, electronic single-access keys, which can be quickly generated on demand, for single users or research institutions. It receives user input data (using the standard SDD format), accepts several key-generation parameters (affecting the key topology and representation), and supports several output formats.

**IKey+**

IKey+ generates single-access keys on demand, for single users or research institutions. It receives user input data (using the standard SDD format), accepts several key-generation parameters (affecting the key topology and representation), and supports several output formats.

IKey+ is freely available (sources and binary packages) at www.identificationkey.fr. Furthermore, it is deployed on our server and can be queried (for testing purposes) through a simple web client also available at www.identificationkey.fr (scratchpads.eu). [Systematics; taxonomy; single-access key; web service; biodiversity informatics.]

**Software for Systematics and Evolution**

**IKey+: A New Single-Access Key Generation Web Service**

THOMAS BURGUIERE*, FLORIAN CAUSSE, VISOTHEARY UNG, AND RÉGINE VIGNES-LEBBE


*Correspondence to be sent to: Thomas Burguiere, Laboratoire Informatique et Systématique, M.N.H.N Département Histoire de la Terre, Bâtiment de Géologie, CP48, 57 rue Curie, 75005 Paris, France, E-mail: thomas.burguiere@upmc.fr

Thomas Burguiere and Florian Causse contributed equally to this article.

Received 2 April 2012; reviews returned 29 June 2012; accepted 25 July 2012

Associate Editor: David Posada

Abstract.—Single-access keys are a major tool for biologists who need to identify specimens. The construction process of these keys is particularly complex (especially if the input data set is large) so having an automatic single-access key generation tool is essential. As part of the European project ViBRANT, our aim was to develop such a tool as a web service, thus allowing end-users to integrate it directly into their workflow.

**ViBRANT is about connecting the people, data and science of biodiversity (ViBRANT - Objectives)**

As studying and monitoring biodiversity remains a strong challenge for biologists, descriptive data management and identification tools are needed to assist them in their daily work. For instance, we developed a complete descriptive data management tool: Xper2 (Ung et al. 2010).

**Our contribution to ViBRANT is the implementation of an identification key generation web service, because we also have a significant experience in developing key generation tools such as MyKey (Gérard and Vignes-Lebbe 2010). The most common identification keys (described in Hagedorn et al. 2010) are the printed single-access keys published in almost all taxonomic revisionary work. But these printed keys may be difficult to use while collecting samples in the field or when consulting incomplete specimens. Hence, electronic single-access keys, which can be quickly generated**

**OUR CONTRIBUTION**

**Approach**

**Web Service**

There are currently several tools designed to generate single-access keys such as, among others, the Delta software suite (Dallwitz et al. 1993), Pankey (Pankhurst 1988), and MyKey. These tools are well established and produce valid identification keys, but are not without drawbacks. For instance, both Delta and Pankey need to be installed on the end-user’s machine and are not cross-platform compatible (Windows, Apple, and Linux). MyKey is a web application, which makes it platform-agnostic, but the end-users cannot submit their own data
set to generate a key (the data set has to be manually uploaded by someone in our laboratory). Furthermore, these 3 tools do not use the biodiversity description standard file format, Structured Descriptive Data (SDD) (Hagedorn 2006), for data input.

Our objective, within the VIBRANT project, was to design a free and open-source key generation service using the SDD file format that can be used by independent users or through the VIBRANT OBOE service (also available through Scratchpads). It was thus decided to develop a web service (implementing both the SOAP and REST web service communication protocols), which is particularly suitable for these two use cases (cf. Fig. 1) and allows the client to build complex workflows, with a high level of automation.

**Architecture**

Because the main purpose of VIBRANT is to provide a platform that is both open source and easily reusable, we organized IKey+ in two parts:

- an Application Programming Interface (API) consisting of 3 distinct modules: the Data Model, that is, the computer representation of the descriptive data, the Input-Output (IO) module, which parses the SDD input files and loads the data in the Model and generates the output files, and the Algorithm, which uses the data contained in the Model to generate a key and returns it through the IO module.
- a Simple Object Access Protocol (SOAP) or a REpresentational State Transfer (REST) Service Layer encapsulating the API, which manages the communication between the client and the API.

This allows potential developers to integrate the service into their workflows and adapt it to their specific needs (e.g., integrating the API in a standalone software). The whole application was developed using the J2EE (Java 2, Enterprise Edition) programming environment.

**Web Service Input Parameters**

IKey+ has several parameters available to the end-user. These parameters allow the end-user to change the topology of the key (e.g., pruning the key, promoting some characters), change the visual aspect of the key, or the output format. A complete list of these parameters can be found in the user documentation, available here: http://www.identificationkey.fr/resources/docs/identificationKeyGeneratorWS_UserGuide.pdf (last accessed 13 August 2012).

**Algorithm**

The single-access key generation algorithm is a recursive depth-first graph construction algorithm. Its arguments are a list of taxa, taxaList and the list of character considered, charList (cf. Supplementary Materials, appendices 1, 2 and 3, available at http://datadryad.org, doi:10.5061/dryad.3ft19). The resulting graph (cf. Fig. 2) is a directed acyclic graph, consisting of non-terminal nodes labelled with a character, edges labelled with the states of the character of the previous node, and terminal nodes labelled with taxa. At each step of the algorithm, the best character among those available is selected by the BEST_CHAR function, which iterates over the available characters and returns the character with the greatest discriminant power.

**Discriminant Power of a Character**

During the last 50 years, estimating a character’s discriminant power has been the central point of key-construction algorithms. Many measurement methods have been suggested, and a review of these methods can be found in Gower and Payne (1975), Pankhurst (1991), and Delgado-Calvo-Flores et al. (2006). In a statistical context, examples include the Bayesian probability and generalized entropy, such as the Shannon entropy used in the ID3 algorithm developed by Quinlan (1986) or the Gini index used in Breiman et al. (1984). In a context where there is no probability associated with the states of the characters, one can use the separation coefficient or the variance as measurement methods of a character’s discriminant power.

In IKey+, the discriminant power of a given character C is calculated by the DPOWER function and is an estimate of C’s ability to differentiate the taxa of the current list. DPOWER is an extension of the Gyllenberg separation factor (Gyllenberg 1963) and
is a measurement of the number of pairs of taxa discriminated by \( C \), which is a generalization of the variance. Indeed, the variance of a variable can be computed by comparing all pairs of values (here, comparing all pairs of taxa description for \( C \)).

With different comparison functions, each adapted to a certain type of character (e.g., a categorical or a numerical character), it is possible to obtain a generalized formula to estimate the discriminant power of a given character, even with polymorphic characters or characters with missing data. For categorical characters, DPOWER can use a binary function (boolean comparison) or other functions such as the Sokal and Michener coefficient (Sokal and Michener 1958) or the Jaccard coefficient (Jaccard 1901). For numerical characters (e.g., a size measured in millimetre), the set of values (i.e., all the values entered for \( C \) for all taxa) is split in two intervals. In order to determine the threshold value separating these two intervals, we consider the list of min and max value of \( C \) for each of the remaining taxa. We then choose from this list the value that separates the remaining taxa into two groups of equal size (±1 taxon). These two intervals are considered as 2 discrete states for the calculation of the discriminant power.

DPOWER iterates over the available taxa (i.e., those that are compatible with the current description), and determines, for each pair of taxa \( T_a \) and \( T_b \), the dissimilarity (i.e., the possibility of discriminating \( T_a \) and \( T_b \) with \( C \)) that is based on the number of common states of \( C \) for \( T_a \) and \( T_b \), the number of states for \( C \) that occur only for \( T_a \), the number of states for \( C \) that occur only for \( T_b \), and the number of states of \( C \) that do not occur for neither \( T_a \) or \( T_b \). The discriminant power is then calculated as

\[
DP = \sum \sum \text{SCOREx}(n_{11}, n_{10}, n_{01}, n_{00}).
\]

Three different SCORE measurements are currently available in IKey+, the Xper coefficient (Ung et al. 2010; Vignes et al. 1989), the Jaccard coefficient, and the Sokal and Michener coefficient.

**BENCHMARKS**

**Tests**

In order to assess the performance of our web service, we conducted a series of tests using the same input file for every test. This file contains a data set representing the subfamily Cichorieae of the plant family Asteraceae with 303 observable characters, 144 taxa, and their descriptions. This data set was generated and curated as an exemplar group for the EDIT project (Hand et al. 2009) [the file is available here: http://www.infosyslab.fr/vibrant/project/test/Cichorieae-fullSDD.xml (last accessed 13 August 2012)]. The aim of these tests was to evaluate the overall performance of the algorithm, to assess the impact of the various parameters available to the end-user on performance, and to ensure that IKey+ would be robust in a production environment. We measured the time necessary for the web service to respond to 100 identification keys generation queries, while counting the number of rejected queries due to CPU overload (IKey+ rejects any query received when the CPU load of the host server is greater than 80%). We ran one reference test (described in Supplementary Materials, appendix 4, doi:10.5061/dryad.3ft19), and several tests with variations on the input parameters, the web service communication protocol used or the parallelization setups. The complete list of performance test setups is available in Supplementary Materials, appendix 5, doi:10.5061/dryad.3ft19. We also measured the time necessary to generate a single key, using the reference test configuration described in Supplementary Materials, appendix 4, doi:10.5061/dryad.3ft19. Finally, we tested the usability of IKey+ from an end-user perspective, when generating the Cichorieae identification key using the web interface. We asked 11 persons who were not
involved in the development of IKey+ to test the web service and the web interface. They were given the SDD formatted Cichorieae data set and were asked to use the web interface to create the identification key. We measured the time needed by each test subject to generate the identification key.

Results

The average length of the paths leading to taxa that were identified by the algorithm is 4.67 steps, with the shortest path being 1 step long, and the longest being 10 steps long. The generation of a single key took roughly 1.8 s. The results of the other tests are shown in Supplementary Materials, appendix 6, doi:10.5061/dryad.3l8l9. The reference test took roughly 50 s to complete, that is, 500 ms per query, which is consistent with the time measured for the generation of a single key, because the reference test uses 4 simultaneous threads to generate the keys. In this test, few queries were rejected due to CPU overload. Among the parameters available to the end-user, only the score method parameter had a significant influence: when using the Sokal and Michener score method or the Jaccard score method, the tests took longer to finish (~80 s instead of 50 s). The score method parameter had no impact on the number of rejected queries. Our tests showed that the communication protocol used (SOAP or REST) had no influence on the performance of IKey+.

Some taxa do not appear in the generated key, due to insufficient data in the input file. Indeed, the Cichorieae data set we used for our tests was created by an external team. This is because some errors were made during creation of the Cichorieae data set: some taxa with unknown data were not specifically marked as “unknown data”, but were left unspecified instead. Regardless, although these ambiguously coded taxa do not appear in the resulting identification key by default, the end-user can choose to have them appear in the key.

When modifying the parallelization of the queries, we observed significant performance variations. As can be expected, when launching 100 queries sequentially (instead of using 4 simultaneous threads launching 25 queries each), the test was 4 times longer, and no queries were rejected. When we augmented the parallelism of the queries (e.g., 25 or 100 simultaneous queries), more queries were rejected (up to 30%). However, when launching 100 simultaneous queries, with a random delay at the beginning of each thread, the results (both in time and number of rejected queries) were comparable to the performance of the reference test. In the usability test, the average time needed to generate the identification key was slightly above 60 s (60.36 s), with the shortest time measured at 26 s, and the longest time measured at 121 s. The complete results of the usability test are available in Supplementary Materials, appendix 7, doi:10.5061/dryad.3l8l9.

Discussion

IKey+ is available and can be installed on any J2EE application server (e.g., Apache Tomcat). It can generate single-access keys using a tree or a flat representation in several output formats (HTML, Wiki, SDD, plaintext, etc.).

Our test showed that IKey+ performs sufficiently well to handle a large data set in a relatively short amount of time and can generate well-optimized key files (average number of steps to identify a taxon: 4.67). This, combined with the web service accessibility, makes it possible to integrate IKey+ in many workflows that might require a fast and automated key generation process (e.g., a batch key generation script). Furthermore, an end-user can use the web interface available at www.identificationkey.fr to quickly generate a customized identification key, using the numerous parameters available (affecting the topology, representation, file format, etc.).

Finally, our tests showed that IKey+ is likely to be robust in a production environment (i.e., many simultaneous queries) as it is able to withstand simultaneous key generation queries (e.g., 4 threads launching 25 consecutive queries). It is also protected from cryptic failure, because we implemented a CPU-load-watching mechanism that automatically rejects a query (with an explicit error message) whenever the CPU load exceeds a given threshold (80%). This prevents a crash of the service, or the generation of incomplete or corrupt key files.

Conclusion

IKey+ is the first key-generation tool available as a web service with standardized input and output formats. Our test showed that IKey+ is able to generate keys rapidly and that it can also be used by an end-user with the web interface. Finally, the modular and open-source nature of IKey+ makes it possible for anyone to reuse its components. For instance, we plan to reuse some components of the API to develop another web service that would provide free-access key identification.

Licensing

As part of the ViBRANT project, IKey+’s source code is freely available and is licensed under the GNU General Public License version 2. It is already available on our google code SVN repository: http://ikey-plus.googlecode.com/svn/trunk/ (last accessed 13 August 2012). It will be actively maintained by our team for the next 2 years.

Supplementary Material

Supplementary material, including Algorithms and appendices, can be found at http://www.sysbio.oxfordjournals.org.
FUNDING
This work was supported by the European Union funded FP7 VI BRANT Project (Contract number RI-261532, Period, December 2010 to November 2013).

ACKNOWLEDGEMENTS
We sincerely thank Gregor Hagedorn (Julius Kühn Institute, Berlin, Germany) and Andreas Müller (Botanical Garden and Botanical Museum, Berlin, Germany) for sharing their knowledge on the SDD format. We are also grateful to Dave Roberts (Natural History Museum, London, UK) for reviewing an early version of the article and providing style improvements.

REFERENCES