Int J Digit Libr (2008) 8:79–90
DOI 10.1007/s00799-007-0032-7

REGULAR PAPER

Stimulating information sharing, collaboration and learning in operations research with libOR

Kris Ven · Kenneth Sörensen · Jan Verelst · Marc Sevaux

Published online: 27 November 2007

Abstract The exchange of data sets within the operations research community suffers from three main issues: (1) data sets are exchanged as plain text files; (2) data sets are offered on centrally managed websites; and (3) the results of applying algorithms to these data sets are unavailable. These issues result in an inefficient sharing of research artifacts. In this paper, we present libOR, a Web-based library of data sets for the operations research community. The organization of libOR is based on the open source and open content approach. The project has three main objectives: (1) stimulate information sharing of research data; (2) increase collaboration to increase scientific advancement; and (3) stimulate learning from approaches undertaken by other researchers in the domain. Early feedback from operations researchers seems to indicate that the advantages offered by libOR are greatly welcomed.

Keywords Operations research · Open source · Collaboration

K. Ven (✉)
University of Antwerp, Faculty of Applied Economics,
Department of Management Information Systems,
Prinsstraat 13, 2000 Antwerp, Belgium
e-mail: kris.ven@ua.ac.be

K. Sörensen
Fellow of the Flemish Fund for Scientific Research,
Centre for Industrial Management, University of Leuven,
Leuven, Belgium
e-mail: kenneth.sorensen@cib.kuleuven.be

J. Verelst
Faculty of Applied Economics, Department of Management
Information Systems, University of Antwerp, Antwerp, Belgium
e-mail: jan.verelst@ua.ac.be

M. Sevaux
LESTER Laboratory Centre de Recherche,
University of South-Brittany, Lorient, France
e-mail: marc.sevaux@univ-ubs.fr

1 Introduction

Since the term *open source software* (*OSS*) was coined for the first time in 1998, it has become an increasingly popular subject. The initial success of open source software was mainly limited to server side applications such as Apache, Bind and Sendmail. However, in the past few years, a lot of progress has been made on the desktop, with applications such as the Mozilla Internet suite and OpenOffice.org. As a result, more and more organizations are currently using—or planning to use—open source software, including many public administrations in Europe [10,29].

Over the years, the open source software phenomenon has attracted a lot of attention from academic research [see e.g., 6,7,16,27]. In general, open source software development is considered to be an alternative software development model. A unique feature is that the software is typically developed by a distributed community of volunteers, which is governed by a peer review process. The open source software development model is therefore a good example of a *peer production model* [2]. Research has identified several advantages of the open source software development model, including a decrease in the number of bugs and increased learning from other project members [19,21,22]. Some organizations, such as Hewlett-Packard, have adopted a slightly modified form of the open source software development model for their internal development and in their collaboration with partners [5].

In recent years, the principles of the open source software development model have been applied beyond the domain of software development. They have been used in various other forms of content creation, such as text, images and multimedia. The term *open content movement* is used to describe the collaborative creation of content over the Internet. This way, the peer production model reaches a potentially much larger community, realizing goals that were impossible for nonprofessionals before [2]. The most famous project in this regard is the Wikipedia[1] project.

The peer production model is also the basis of academic research [2]. Some authors have noted that certain domains of academic research have not maximized the potential of electronic communication channels such as the Internet, and that some processes can be improved by using methods from the open source movement [2,11,15,24].

In this paper, we discuss how the open source software and open content model may contribute to the advancement of academic research in the operations research domain. Operations research applies scientific methods to aid in resolving various problems in business, logistics and many other domains. New algorithms for optimization problems are tested by applying them to a standard collection of data sets, so that the performance of various algorithms can be compared. Currently, the exchange of these research artifacts suffers from a number of issues which we aim to solve. These issues are not as much from a technical nature, because the research artifacts are small and relatively easy to standardize. The main challenge, however, is to create an environment that stimulates information sharing, collaboration and mutual learning within the operations research community in order to increase the proliferation of scientific advances and enhance the reliability of scientific results. We therefore developed an IT infrastructure whose organization is inspired by the open source and open content movement. This resulted in libOR: a Web-based library or repository of optimization problem data sets for operations research. The content of libOR is created collaboratively by the operations research community.

The rest of the paper is structured as follows. Section 2 describes the operations research domain, and the issues this domain is currently facing. In Sect. 3 the libOR library is introduced. We first take a look at the technical structure of the library and continue with the processes underlying the library. Section 4 discusses the implementation of the IT platform that underlies libOR and that is built using open source software. In Sect. 5, we discuss the concrete advantages the open source/open content-based approach of libOR offers to operations researchers. Using the competency rallying theory [14], we also evaluate the capability of libOR to become successful in the future. Finally, our conclusions are presented.

2 Operations research

Operations research (*OR*) can be defined as the application of scientific methods (mathematics and statistics) to aid in the process of decision making. Operations research originated in World War II, when scientists realized that the enormous logistical operations could be further optimized by applying mathematical methods. This is how the name *operations* research originated. After the war, similar methods were developed and applied to various problems in business and industry. The term *management science* is used interchangeably when working on business management problems. Practical applications of operations research are the design of telecommunication networks, routing planning for the supply of a distribution chain and class room scheduling in universities. As can be seen from these examples, the goal is typically to improve or optimize the performance of processes. The operations research domain is constituted of several subdomains such as optimization, simulation, statistics, graph theory and queuing. The libOR project is specifically targeted to research in the optimization subdomain.

2.1 Optimization

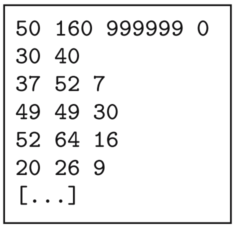
Optimization constitutes a large part of the broader discipline of operations research. In optimization, algorithms and heuristics are developed to find a *solution* which maximizes or minimizes the value of an *objective function* subject to some *constraints*. The difference between a heuristic and an (exact) algorithm is that the former does not offer the guarantee that the solution it finds is *optimal* (i.e., the best possible solution). A famous example of an optimization problem is the traveling salesman problem (TSP). In this problem, a set of spatially distributed "cities" are given, and the distance between each pair of cities is assumed to be known. The objective of this problem is—starting from a given city—to find the order in which to visit all cities and return to the starting point in such a way that the total distance traveled is minimized. Although the problem is very easy to state and has many practical purposes in such diverse areas as logistics and genome sequencing, it is very hard to solve because the computation time of all known algorithms rises exponentially with the size of the problem. Recently, state-of-the-art methods succeeded in finding the optimal solution to a TSP consisting of all 24,978 cities of Sweden. The computation required to find this solution was performed on a large cluster of parallel computers, but would have required almost 85 years on a single Intel Xeon 2.8 GHz.[2]

The excessive computational time required to find the optimal solution to a large number of interesting problems has given impetus to the development of heuristics, which attempt to quickly identify a solution that is "good enough". More recently, the development of more advanced heuristics, commonly known as *metaheuristics*, has received increasing attention.[3]

[1] http://www.wikipedia.org

[2] http://www.tsp.gatech.edu/sweden

[3] See e.g., http://webhost.ua.ac.be/eume



```
50 160 999999 0
30 40
37 52 7
49 49 30
52 64 16
20 26 9
[...]
```

Fig. 1 Extract from a capacitated vehicle routing (CVRP) instance (source: OR Library)

To test the performance of newly developed algorithms and (meta)heuristics for a specific problem, they are generally applied to a suite of *test problems*. The importance of having high-quality test problems readily available cannot be underestimated. This allows for the comparison of the performance of the new algorithm with that of the previously developed algorithms. It is therefore essential that these data sets are shared between researchers in order to facilitate scientific advancement. The current way in which these data sets are distributed, however, leaves much room for improvement.

2.2 Issues

We identified three main issues with the current distribution of data sets.

A first issue is that test problems are commonly distributed as plain text files. An example of an extract of a data set for a vehicle routing problem can be seen in Fig. 1. The use of plain text files has three disadvantages. A first disadvantage is that such a data set usually only contains a set of numbers, without any semantic information of any kind. This makes the file difficult to understand, even for an experienced operations researcher. In some cases, a limited explanation of the structure of the file is offered in the file, or in a separate file, but this does not prevent many programming errors that are very difficult to trace. An example of such an error is reading the wrong numbers into the wrong variables. The structure of data sets, even those covering the same domain such as vehicle routing, may also have a slightly different structure depending on the author. This requires the researchers to either rewrite the input parser for each data set, or convert the data set into a standard format, either manually or by writing their own conversion tool. A second disadvantage of using plain text files is that there is no way to identify a file as a valid instance of a data set for a specific problem instance. This makes it difficult to judge whether an observed problem should be attributed to the algorithm or to the data set. A third disadvantage is technical differences between operating systems such as end-of-line conventions or character encodings, which are troublesome for many researchers.

A second issue with the current way of exchanging data sets is the fact that test problem suites are scattered across the Internet. The closest to a central repository of problem data sets is the so-called *OR library*[4] which was first described in Beasley [1]. This database is maintained by a single person, making it very slow to respond to current developments. This disadvantage is confirmed by Gendreau et al. [9] who found in a survey on metaheuristics for vehicle routing that authors do not always test their approaches on the newest data sets and therefore often report outdated information. Moreover, the OR library is far from being a central library for all data sets. Data sets for the TSP, for example, are not included and are distributed through the so-called TSPLIB.[5]

A third issue is that none of the libraries mentioned above include the results obtained from applying different algorithms to the various data sets. An operations researcher who has developed a new algorithm for a certain type of problem is therefore forced to undertake a time-consuming literature review, in search of the performance of previously developed algorithms. The literature is however far from complete, as there is some truth to the often-heard complaint that a paper in optimization is only considered publishable if the heuristic developed in it succeeds in finding at least one solution that is better than the best solution previously known. Operations researchers are therefore likely to undertake efforts that have already been tried before by other researchers and have proved to lead to no improvement. The literature in general does however not contain any record of these attempts.

2.3 Operations Research and Open Source Software

A related open source-based initiative in operations research is the *COmputational INfrastructure for Operations Research* (*COIN-OR*)[6] [18]. In operations research, the mathematical representation of the algorithms developed is published in academic journals. The implementation of these algorithms, in source code form is however not published. This leads to important inefficiencies as researchers must for example re-implement existing algorithms due to the lack of source code availability. As a result, knowledge sharing between researchers is limited [18].

The aim of the COIN-OR project is to make the implementation of various algorithms available to researchers in the operations research community. The implementations submitted to COIN-OR are licensed under the terms of the Common Public License (CPL), an open source license developed by IBM. This way, the same implementation can be shared and further improved upon by other researchers. The COIN-OR project consists of several subprojects which develop algorithms for a specific type of problem, and has attracted a considerable quantum of interest.

[4] http://www.brunel.ac.uk/depts/ma/research/jeb/info.html

[5] http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95

[6] http://www.coin-or.org

The COIN-OR project is however fundamentally different from libOR. Although both projects were initiated due to a lack of collaboration between researchers, both projects undertake a different approach in tackling these issues. COIN-OR uses a centralized website, divided in subprojects, where users can download code. LibOR on the other hand, provides an interactive environment with a different underlying philosophy. Moreover, the COIN-OR project provides a community-driven platform for the sharing of programs and source code, whereas libOR provides an interactive platform for the sharing of research artifacts such as data sets and results.

3 LibOR

In Sect. 2.2, we mentioned the different issues the operations research domain is currently facing. In order to tackle these issues, we started the *libOR* initiative.[7] The primary aim of this project was to stimulate information sharing within the operations research community, based on an open source/open content approach. We can divide the issues of Sect. 2.2 in to two categories: technical issues and organizational issues. The technical issues refer to *which information* is currently created and shared amongst researchers. The organizational issues refer to *how information* is created and shared within the operations research community. We first discuss the technical solution offered by libOR, devoting some attention to both the database structure and the way data sets are stored. Next, we proceed with the discussion of the collaborative environment that libOR provides.

3.1 Technical solution

3.1.1 Database structure

The database structure of libOR is shown in Fig. 2. The library itself containing the data sets is organized around a shallow hierarchical structure. At the top level, libOR consists of a number of *problem categories*, each of which in turn consists of a number of *problems*. A problem category is a general container for several related problems such as routing or scheduling. A *problem* is defined as a specific type of optimization problem, e.g., "the capacitated vehicle routing problem with time window constraints", or "the total weighted tardiness single-machine scheduling problem with release dates". A problem may belong to more than one category.

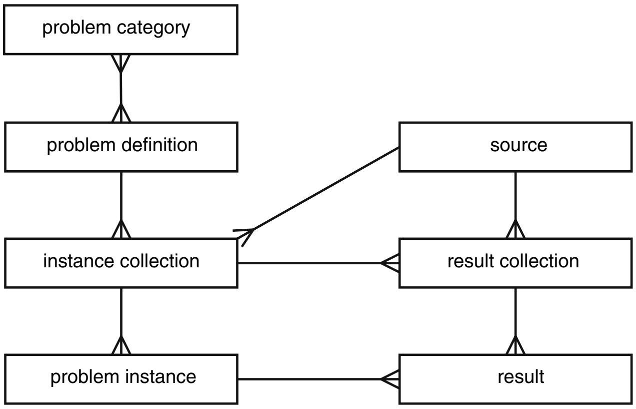


Fig. 2 Entity-relationship diagram of the database structure of libOR

A problem may have any number of associated *instance collections*. An instance collection is a set of related *instances*, for example originating from the same source (e.g., the Christophides et al. [3] instances for the CVRP). The term *instance* refers to a single data set in XML format, on which to test an optimization algorithm. All data sets (XML files) of an instance collection must conform to the XML Schema Definition defined for the corresponding problem definition.

Besides problem instances, libOR also contains results. This facilitates the comparison of the performance of different algorithms and heuristics on the same data set. Results are grouped into result collections that correspond to an instance collection. The format of a result (or a solution) is problem-dependent and can include data such as the best value obtained and the total computation time.

Finally, bibliographic information is also stored on libOR. A "source" is a description of the origin of either an instance collection or a result collection. This can for example be a website, a published article or a working paper. Sources can be re-used, i.e., assigned to several instance or result collections.

3.1.2 Data format

The technical issues concern the data format that is used for the storage and exchange of data sets. As previously mentioned in Sect. 2.2, the use of plain text files to store data sets has three important disadvantages: (1) data sets contain no semantic information; (2) checking the validity of the data set is impossible; and (3) technical differences between operating systems. In order to address these disadvantages, we opted for XML files for data storage.

The first advantage of XML is that XML files contain *semantic information* within the file. Figure 3 shows the same data set as in Fig. 1, but this time in XML format. It is obvious—even for a researcher who is not all too familiar with the specific type of problem—that this data set is much easier to read and understand. This will prevent logical programming errors such as reading in the wrong information into the wrong variable. It is also possible to perform a first, manual

[7] http://webhost.ua.ac.be/libor

```
<?xml version="1.0" encoding="UTF8"?>
<vrp xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:noNamespaceSchemaLocation="cvrp.xsd">
  <vehicles>
    <number>50</number>
    <capacity>160</capacity>
    <maxdistance>999999</maxdistance>
    <dropcost>0</dropcost>
  </vehicles>

  <depot>
    <depotx>30</depotx>
    <depoty>40</depoty>
  </depot>

  <customers>
    <customer>
      <id>1</id><x>37</x><y>52</y><demand>7</demand>
    </customer>
    <customer>
      <id>2</id><x>49</x><y>49</y><demand>30</demand>
    </customer>
    <customer>
      <id>3</id><x>52</x><y>64</y><demand>16</demand>
    </customer>
    <customer>
      <id>4</id><x>20</x><y>26</y><demand>9</demand>
    </customer>
  </customers>
</vrp>
```

Fig. 3 The capacitated vehicle routing extract in XML

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:element name="vrp" type="vrpType"/>
 <xsd:complexType name="vrpType">
  <xsd:all>
   <xsd:element name="vehicles" type="vehiclesType"/>
   <xsd:element name="customers" type="customersType"/>
   <xsd:element name="depot" type="depotType"/>
  </xsd:all>
 </xsd:complexType>
 <xsd:complexType name="vehiclesType">
  <xsd:all>
   <xsd:element name="number" type="xsd:unsignedInt"/>
   <xsd:element name="capacity" type="xsd:unsignedInt"/>
   <xsd:element name="maxdistance" type="xsd:unsignedInt"/>
   <xsd:element name="dropcost" type="xsd:unsignedInt"
                minOccurs="0"/>
  </xsd:all>
 </xsd:complexType>
 <xsd:complexType name="depotType">
   <xsd:sequence>
     <xsd:element name="depotx" type="xsd:int"/>
     <xsd:element name="depoty" type="xsd:int"/>
   </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="customersType">
  <xsd:sequence>
   <xsd:element name="customer" minOccurs="0"
                maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="id" type="xsd:int"/>
      <xsd:element name="x" type="xsd:integer"/>
      <xsd:element name="y" type="xsd:integer"/>
      <xsd:element name="demand" type = "xsd:unsignedInt"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:schema>
```

Fig. 4 Capacitated vehicle routing XML schema file

evaluation of the validity of the data set. If the number of vehicles for example would be a negative number, it is clear that the data set contains an error. Manually checking the validity of a whole data set would however be a tedious task.

A second advantage of XML is that it allows the automatic validation of an XML file against an external file that describes the general structure of that file. In the past, Document Type Definitions (DTD) were used, but they have been superseded by XML Schema. Although XML Schema is a more complicated language than DTD, it offers far more advanced features than DTDs. Therefore, we have opted for using XML Schema. In libOR, each type of problem (problem definition) is assigned an XML Schema Definition (XSD) file. This XSD describes the format to which all concrete problem instances (data sets) that fall under this problem definition must comply. The XML Schema Definition for the CVRP data set is shown in Fig. 4. Based on this XSD, a validating XML parser can automatically check whether the XML file is a valid problem instance for a given problem definition. Many tools are currently available. There are commercial products such as *XMLSpy*,[8] but there are also several open source alternatives available. Most well known is Xerces[9] from the Apache Software Foundation which is available for Java, C++ and Perl. Another command line driven program is *xmllint*,[10] which is part of the XML parser and toolkit of the Gnome project, a graphical desktop environment for Linux. Both Xerces and xmllint are available for Windows as well as Linux and Unix systems. As will be shown in Sect. 3.2, the Xerces XML parser is integrated within libOR to allow the on-line validation of data sets.

A third advantage of using XML is that XML files are highly interchangeable between different computer platforms. This means that platform-specific storage technicalities such as end-of-line conventions and character encodings are no longer troublesome for the end-user. When using XML, the XML parser will automatically detect these encodings and handle them accordingly. Since the handling of the encodings is done by the parser library, the user is freed from this task.

XML is also an appropriate choice, given the type of data at hand. As the name implies, XML (the Extensible Markup Language) is very flexible and can be easily extended. This allows the creation of custom tags, specific to a certain problem definition. As a result, it is not required to use a fixed format for each type of problem. Moreover, XML files can have a hierarchical structure, which is convenient for most

[8] http://www.altova.com/products_ide.html

[9] http://xml.apache.org

[10] http://xmlsoft.org

types of problems. The data set shown in Fig. 3, for example, has a clear hierarchical structure.

The switch from using plain text files to XML files means that data sets cannot be read and processed directly. Instead, data sets must first be parsed. The output of this parsing process, a data structure residing in computer memory, will then be available for further processing by the algorithm. This will certainly require a change for most researchers. We will return to this problem in Sect. 5.2. There are however many XML parsers available, open source or commercial. Having to parse the data set instead of just reading it, may also have a positive result. By using external libraries that handle the input, the researcher is freed from having to re-implement the input routines each time, so that he/she can focus on the development and testing of the algorithm itself.

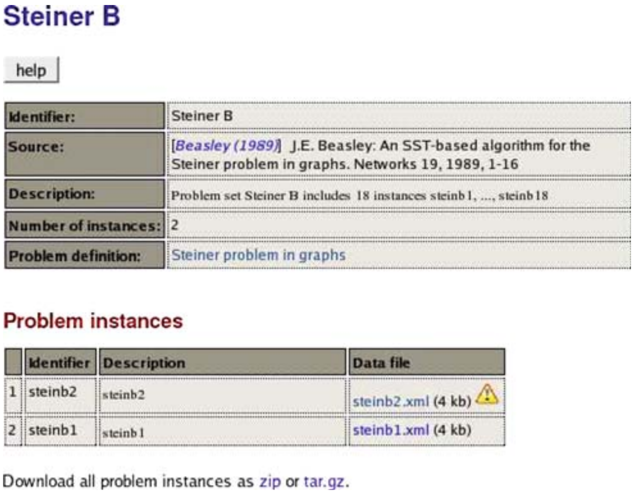
3.2 Collaborative environment

Besides the technical improvements libOR offers to operations researchers, it also provides a collaborative environment in which content can be created. The processes and practices behind libOR are similar to those used in open source and open content projects. As suggested by some authors [see e.g., 11,23,24], knowledge creation and sharing in academic research can be leveraged by using a more open approach. All information that is submitted to libOR is publicly available for anyone, even for people who are not active in operations research. In the remaining section, we will focus on how content is created in libOR. We will also indicate which open source practices are used in each task.

3.2.1 Adding data sets

The central feature of libOR is that data sets can be produced by and offered to the operations research community. As mentioned in Sect. 2.2, there are fundamental issues when using a centralized approach, which results in outdated and incomplete libraries. However, a collaborative approach has proved to be very powerful in projects such as Wikipedia and the NASA Clickworkers project[11] [see also 2]. Therefore, we used an open community model, in which every researcher within the OR community can submit new content to the libOR website and help in the further development of the library.

In order to guarantee the correctness of the information on libOR, the newly added data sets are not immediately shown, but are subjected to a *peer review* process, as it exists in open source projects [see e.g., 6,20]. For each problem category, one or more responsibles are assigned. Once a new data set is submitted to the library, it is assigned the "pending" status. The validity of the data set is automatically checked by Xerces against the respective XML Schema Definition. The output of the parsing process is also saved in the database. Next, the corresponding responsible is alerted and invited to review the new data set. Figure 5 shows a Web page containing the information of a new instance collection. If one of the data sets contains an error, a warning icon is displayed next to the file name with a link to a more detailed error description (see Fig. 6). If required, the responsible can modify the data set as required when there are parsing errors, contact the contributor to ask him to modify the data set, or decide to reject the data set altogether when the data set is of insufficient quality or not appropriate for the problem definition. Once a data set has been approved, it is shown on the website and everyone can view and download it. The peer review process is however not limited to the responsible: once the data set is publicly available, everyone can report any remaining errors to the responsible, so that appropriate action can be taken. This principle builds on Linus' Law, namely that the more people are reviewing code or content, the quicker errors are detected. In order to aid the peer review process, it is also encouraged that contributors provide a bibliographic reference to where the data set or result set was first published. In most cases, this will be a paper in a conference proceedings or journal.



Steiner B

Identifier:	Steiner B
Source:	[Beasley (1989)] J.E. Beasley: An SST-based algorithm for the Steiner problem in graphs. Networks 19, 1989, 1-16
Description:	Problem set Steiner B includes 18 instances steinb1, ..., steinb18
Number of instances:	2
Problem definition:	Steiner problem in graphs

Problem instances

	Identifier	Description	Data file
1	steinb2	steinb2	steinb2.xml (4 kb) ⚠
2	steinb1	steinb1	steinb1.xml (4 kb)

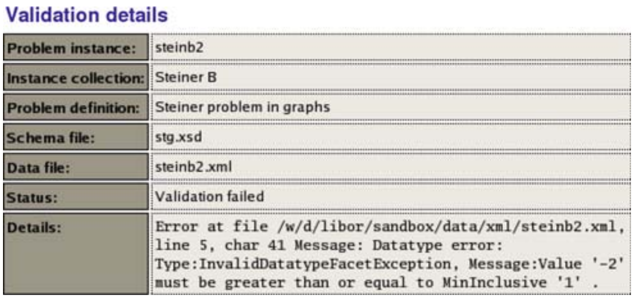
Download all problem instances as zip or tar.gz.

Fig. 5 Web page showing the contents of an instance collection

3.2.2 Proposing new problem definitions

To make the future expansion of libOR possible, people can make suggestions for new problem definitions. However, we wanted to avoid a proliferation of categories and problems. Moreover, the domain of possible problems is quite stable, hence future additions will be rather limited. Therefore, the submission of new problem definitions is also subjected to a peer review process. A contributor must provide a precise definition of the type of problem, a proposal for the

[11] http://clickworkers.arc.nasa.gov/top



Validation details

Problem instance:	steinb2
Instance collection:	Steiner B
Problem definition:	Steiner problem in graphs
Schema file:	stg.xsd
Data file:	steinb2.xml
Status:	Validation failed
Details:	Error at file /w/d/libor/sandbox/data/xml/steinb2.xml, line 5, char 41 Message: Datatype error: Type:InvalidDatatypeFacetException, Message:Value '-2' must be greater than or equal to MinInclusive '1' .

Fig. 6 Web page showing the details of a parsing error

XML Schema Definition and the required output variables of the algorithm (e.g., processor type used, the computing time and the objective function value). Once a new suggestion is received, the appropriate responsible will open a thread on an integrated discussion forum. In this thread, every operations researcher can provide comments on the new problem definition. Once there is a consensus, the problem definition can be published. After that, data sets can be submitted to the new problem definition.

It is important to note that the responsibles on libOR are unpaid volunteers. This is once again quite similar to the traditional open source model in which most contributors are volunteers [6]. Since there are no financial incentives for accepting the role of responsible, the responsible's primary motivations are peer recognition from other members in the community, intrinsic motivations, the pleasure of creativity (also a driving force in academic research) and altruism. These motivations are similar to the motivations of OSS developers [6,17].

The selection of responsibles follows a *meritocracy* model. This meritocracy model is often found in open source projects, such as the Apache Software Foundation [6,8]. This means that although anyone in the OR community is invited to volunteer, the assignment is dependent on previous efforts. In a first phase, the selection is based on the amount and quality of research undertaken by the volunteers in the operations research domain. In a second phase, we will also take into account their level of contribution to libOR. We hope that these criteria will help in appointing responsibles whose judgment will be trusted and accepted by other members in the community.

3.2.3 Submitting results

A last feature of libOR is that solutions and results obtained by applying an algorithm to a specific instance collection, can be submitted to libOR, hence resolving the third issue mentioned in Sect. 2.2. This allows researchers to compare different approaches (algorithms) on the same dataset. Unlike the previous two actions, results submitted by contributors are immediately shown in the library, without any intervention of the responsible. Verifying each submitted result collection would result in too much work for the responsible, while it is more important to verify the quality and validity of the data sets. Moreover, we trust the community members to submit correct information to libOR. Continuously submitting false content would degrade their position and respect in the community. Nevertheless, the peer review process is still carried out by the community. Researchers who find errors or inconsistencies in the reported results, can alert the responsible or contact the contributor directly in order to clarify their results.

4 Implementation

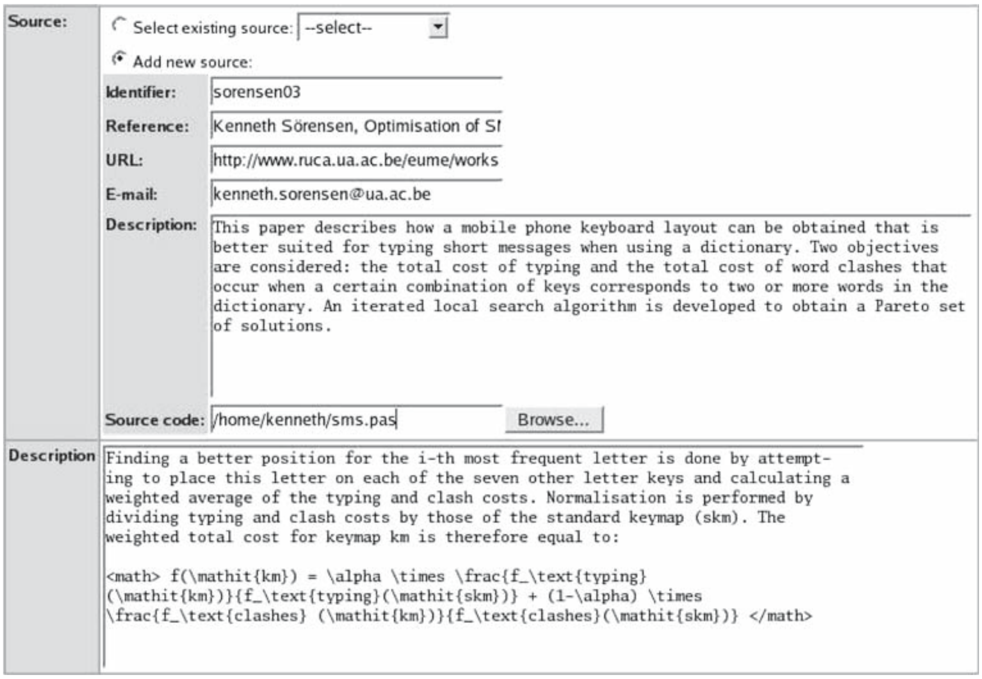
During the development and implementation of libOR, we made exclusive use of open source software. This decision was motivated by several reasons. First, thanks to the availability of many reliable open source packages, existing software could be reused which sped up the development process considerably. Second, the open source licenses allow the modification of existing source code. This allowed us to modify some components, or use only a part of a larger open source package so that it would better integrate with the rest of libOR. A final, but quite important reason, is that open source software is available free of charge. This means that development and operating costs could be kept to a minimum. This, together with the hosting support by the University of Antwerp, allows us to offer libOR and all the data it contain free of charge.

The libOR software runs on a LAMP (Linux, Apache, MySQL, PHP) platform. This is generally known to be a reliable platform, with which we had positive experiences in previous projects. Another advantage of this platform is that PHP is a relatively easy-to-learn language, which makes it easier to attract new developers.

Three other open source software packages were integrated with the libOR software. As already mentioned, we use Xerces to verify the validity of uploaded XML data sets and to provide the responsible with an overview of possible errors in the data set. This way, a responsible or user can immediately determine whether the data set is valid or not, without having to download the data set and run the validator on his local machine (see Figs. 5 and 6).

In most description fields in entry forms on libOR, it is possible to use LaTeX formulas. This is convenient as operations research makes extensive use of mathematical formulas. Although MathML is slowly becoming more accepted, displaying MathML is only possible in newer browsers, or requires the installation of a separate plugin. Therefore, we opted for *Texvc* (*TeX validator and converter*)[12], a program

[12] http://en.wikipedia.org/wiki/Texvc



Source:

○ Select existing source: --select--

● Add new source:

Identifier: sorensen03

Reference: Kenneth Sörensen, Optimisation of SI

URL: http://www.ruca.ua.ac.be/eume/works

E-mail: kenneth.sorensen@ua.ac.be

Description:

```
This paper describes how a mobile phone keyboard layout can be obtained that is
better suited for typing short messages when using a dictionary. Two objectives
are considered: the total cost of typing and the total cost of word clashes that
occur when a certain combination of keys corresponds to two or more words in the
dictionary. An iterated local search algorithm is developed to obtain a Pareto set
of solutions.
```

Source code: /home/kenneth/sms.pas

Description

```
Finding a better position for the i-th most frequent letter is done by attempt-
ing to place this letter on each of the seven other letter keys and calculating a
weighted average of the typing and clash costs. Normalisation is performed by
dividing typing and clash costs by those of the standard keymap (skm). The
weighted total cost for keymap km is therefore equal to:

<math> f(\mathit{km}) = \alpha \times \frac{f_\text{typing}
(\mathit{km})}{f_\text{typing}(\mathit{skm})} + (1-\alpha) \times
\frac{f_\text{clashes} (\mathit{km})}{f_\text{clashes}(\mathit{skm})} </math>
```

Fig. 7 Entry form for a new instance collection

that is capable of transforming LaTeX formulas into HTML, MathML or PNG (Portable Network Graphics) images. Texvc is a small program that is part of Mediawiki, the software that is used to support Wikipedia. In libOR, Texvc is used to transform the LaTeX formulas into PNG images. In most description fields, LaTeX formulas may be entered by placing them between <math> and </math> tags. Figure 7 shows how LaTeX formulas can be entered in a form on the website, and Fig. 8 shows how the description field will look like after rendering by Texvc.

We also decided to add a discussion forum to libOR to support the discussion on new problem definitions, and offer a place where operations researchers can take part in a general discussion on libOR to support the collaborative content creation. Many advanced discussion forums written in PHP are currently available. We decided to use Phorum[13], a relative simple discussion forum that can be easily integrated with the rest of libOR.

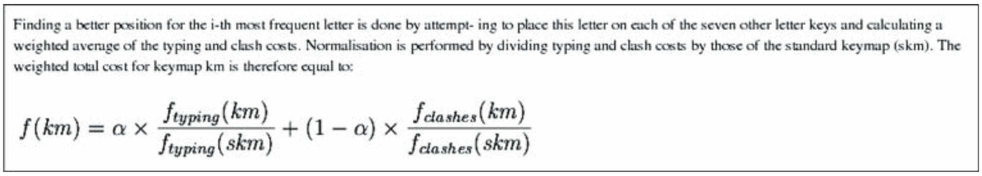
Early in the development we decided to release the source code of libOR under an open source license. To differentiate the source code of libOR from the actual libOR library, we named the code base of libOR "*ORLi*". Our main motivation for releasing ORLi under an open source license was to emphasize the open character of libOR. The GNU GPL was chosen as license, because this license offers the most freedom to the creator of and contributors to the project. We hope that by making the source code open to everyone, we will realize two advantages. First, it may speed up the development process as each member in the community can suggest and implement new functionality without having to rely on a central entity to perform this task. The new functionality can later be added to the production version of libOR. It remains a fact however that most operations researchers do not have programming experience in Web-based environments. Hence, it remains to be seen to which degree libOR will attract other developers. The COIN-OR project that was discussed earlier however, seems to indicate that open source projects in operations research can be successful. A second possible advantage is that, following Linus' Law, even if operations researchers do not take part in the active development of libOR, community members can still report bugs and help improve the overall quality of libOR.

We do not expect that competing versions of libOR using the ORLi source code will appear within the operations research domain, as this would decrease the value of the product to the community. In that case, data sets would still be distributed over many different libraries. Additionally, in open source projects there exists a certain social barrier against forking (the creation of a modified version of an open source project, without contributing back to the original project).

5 Experiences

In this section, we will summarize the concrete advantages that libOR offers and discuss the capabilities the project possesses.

[13] http://www.phorum.org



Finding a better position for the i-th most frequent letter is done by attempt- ing to place this letter on each of the seven other letter keys and calculating a weighted average of the typing and clash costs. Normalisation is performed by dividing typing and clash costs by those of the standard keymap (skm). The weighted total cost for keymap km is therefore equal to:

$$f(km) = \alpha \times \frac{f_{typing}(km)}{f_{typing}(skm)} + (1-\alpha) \times \frac{f_{clashes}(km)}{f_{clashes}(skm)}$$

Fig. 8 The description field as rendered by Texvc

5.1 Advantages

LibOR offers several important benefits to operations researchers, since its features address the issues mentioned in Sect. 2.2.

A first advantage follows from the use of XML as data format. Thanks to this data format, semantic information is included within the data set and validation against an XML Schema Definition is possible. Both features reduce errors in creating and processing data sets. As a result, researchers will lose less time with programming technicalities. More reliable input evidently also leads to more reliable results. Additionally, XML is platform independent and can therefore easily be shared between different operating systems. Technicalities such as end-of-line conventions are handled by the XML parser.

The second advantage is that libOR aims to be a central repository for all operations research problems. The value of central repositories such as OR library has already been demonstrated in the past. This project however aims to combine all these repositories. Moreover, the collaborative environment which is the basis of libOR should facilitate cooperation among operations researchers and increase rapid proliferation of scientific advances. This also means that the list of data sets can be updated more often, and new developments in operations research are followed. Currently, an operations researcher who wishes to work on a problem he/she has not worked on before faces the tremendous task of searching the literature for every paper in which this problem is tackled. LibOR may dramatically improve upon this situation as it provides the operations researcher with a "one-stop" repository. Thanks to the improved and wider availability of data sets, operations researchers will be more likely to test their algorithms on a wider, and more recent selection of data sets. This will lead to more reliable and robust results.

The third benefit is the easy comparison of several algorithms, thanks to the inclusion of the results of applying algorithms to the data sets. This will make it easier to assess the real performance of the algorithm. Moreover, the submission of results is not only limited to "successful" attempts. As mentioned earlier, journals in optimization tend to publish only papers which present methods that improve upon the best-known algorithms. The result of this is that failed attempts of applying a specific method to a specific problem are never released. This in turn leads to the inefficient situation that every researcher goes through a frustrating "learning period" in which he/she makes the same mistakes many have made before. The possibility to report the results of failed attempts may reduce the length of this learning period.

5.2 Project capabilities

In this section, we will discuss whether libOR has the potential to become a successful project. We will base our discussion on the *competency rallying* (*CR*) theory, applied to open content projects. The CR theory was developed by Katzy and Crowston [12] which lists four capabilities a successful virtual organization must possess. The CR theory integrates the resource-based theory of the firm with other theories on networking, virtual organizations and dynamic capabilities. The CR theory was first tested in a virtual organization in the precision manufacturing industry [13], and was further validated and generalized to the open source context by applying the theory on a large set of open source software projects hosted on SourceForge [4]. Keats [14] has further extended the CR theory by giving an interpretation of the competency rallying theory in the context of open content projects. This interpretation is shown in Table 1. We will apply each of these four capabilities to the libOR project.

Identification and development of individual competencies This capability states that projects should be started where there is a critical mass of expertise available to support the project. In the case of libOR, it is evident that the operations research community is the target audience, and that this community possesses the knowledge required to create new data sets and problem definitions. Moreover, the operations research community is quite large, which is confirmed by the number of journals and conferences in this domain. We also have already attracted a considerable number of volunteers working on libOR.

However, operations researchers in general do not have a computer science background which may be an issue when using libOR, given the use of XML. In our experience, this has been one of the problems encountered when inviting researchers to act as a responsible. The fear of having to convert all plain-text data files in a category to XML may seem

Table 1 Capabilities of successful open content projects [14]

Capability	Proposition	Open content interpretation
Identification and development of individual competencies	The more available the required competencies, the more successful the open source software project	Open content projects should start subject areas where there is a critical mass of available expertise, and such projects will achieve more than those where expertise is limited
Identification of market opportunities	The more readily developers can recognize the needs and problems addressed in the project, the more successful the open source software project	Open content authors should include as many of the content users as possible; learners should contribute the identification of opportunities for improvements to content and related learning objects; ample opportunities for feedback should be provided
Marshalling of competencies	The more quickly and accurately competencies can be marshalled, the more successful the open source software project	Open content projects will succeed better if they are led by a champion who is well-connected in the discipline for which content is being developed
Management of a short-term co-operative effort	The greater the ability to manage short-term cooperation, the more successful open source software project	Open content projects should "hit the ground running" under the leadership of a known and respected champion who understands the importance of communication to the open content process.

a daunting task. Creating the corresponding XML Schema Definitions probably poses an even greater challenge. Further, there seems to be an unjustified concern that writing programs which work with XML input files is much more difficult than writing programs which read the corresponding plain text files.

In order to overcome this barrier, we will provide basic and intermediate information on XML, together with concrete examples on how data sets and XML Schema Definitions need to be created. XML tool support will also facilitate these tasks. Commercial tools such as Oxygen[14] and XMLSpy, for example, provide a graphical interface to construct XML Schema files. This certainly reduces the technical knowledge required for constructing XML Schema Definitions. We are therefore confident that in time the knowledge of XML will increase, and that the use of plain text files for the exchange of data sets in optimization will fade away. The researcher may be faced with a steep learning curve at first, but we are convinced that the benefits (especially the validation of data files) greatly outweigh the drawbacks.

Identification of market opportunities This capability means that a large audience must be reached, and that opportunities for improvement and feedback must be indicated. Through the dissemination of our work, we aim to reach as much of our target audience as possible. The libOR initiative was first presented at the *Workshop on Real-Life Applications of Metaheuristics* in December 2003 [25]. In October 2004, it was presented at the INFORMS annual meeting in Denver, the largest conference on operations research [26]. It was also presented at *The First International Conference on Open Source Systems* in July 2005 [28]. On each occasion, libOR attracted considerable interest. Feedback obtained from these efforts indicate that operations researchers indeed recognize the potential of libOR.

Thanks to the open content and open source based approach, we are also convinced that the opportunities for the community to offer feedback are maximized. This is illustrated by, for example, the open peer review process and the availability of the libOR source code under an open source license.

Marshalling of competencies This principle requires that competencies in the project must be marshalled, and that it should be led by a *champion* who is well connected to the community. We try to achieve this by assigning part of this task to the category responsibles. These responsibles are chosen based on their past efforts in the field, which will make them more accepted by the community. Additionally, two of the authors are actively involved in the operations research domain and founded the EU/ME Working Group on Metaheuristics.

Management of a short-term cooperative effort This rule highlights the need to realize important growth in the starting phase of the project, to maximize the chances on success. Currently, there is a production version of libOR available that is well tested, and there is a sufficient number

[14] http://www.oxygenxml.com

of volunteers who act as responsible. These responsibles are working on the conversion of the existing data sets to XML format, and the creation of XML Schema Definitions. This way, we will be able to offer an initial quantum of information that can kickstart the project.

Apart from the production version, there is a "sandbox" version of libOR available. This is an identical copy of the production version, but it contains no production data. In the sandbox, users can experiment and get familiar with the features of libOR. This sandbox is quite active and lowers the entry barriers for potential volunteers.

The ultimate success of libOR is however heavily dependent on the operations research community. Therefore, the adoption of libOR by the community is subject to *network externalities*: the advantages of joining libOR are larger when the user base is larger. It remains therefore important to be able to attract a critical mass of users early in the project.

In order to realize the goals of the libOR project, operations researchers must be convinced to contribute their research results to libOR. This is however a rather different approach than the traditional publication of research, since in libOR not only the final results are published, but also the intermediary products (e.g., data sets). This may require a cultural change for some researchers. Schweik et al. [24] also found that some researchers did not want to share their data with non-researchers when using their Open Research System, leading to the development of a "private" website that is only accessible for researchers. So far, we have found no indication that a similar evolution might take place with libOR. A possible explanation for this might be that a culture of sharing already exists in the operations research community to some degree. Moreover, the data sets that are shared in libOR are quite useless for non-researchers. We therefore expect that there will be a large wave of early adopters. This, combined with the many benefits that libOR offers to operations researchers, and the fact that we fulfill the four criteria of the competency rallying theory, leads us to expect that libOR is likely to become successful.

6 Conclusion

In this paper, we have presented libOR, which aims to be a one-stop repository for operations researchers. The project was realized by applying open source and open content methods and processes to the operations research domain in order to resolve some important issues that exist in operations research nowadays.

This paper makes three main contributions. First of all, it illustrates that the principles of the open source movement have a large potential application domain, even outside software engineering. The libOR project applies the open source methods in one of the domains of academic research, namely operations research. Another open source project, COIN-OR already exists in the operations research community with the aim of producing algorithms and software libraries under an open source license. The libOR project is however fundamentally different and is the first project within operations research to publish content based on the open source and open content principles.

The second contribution is that libOR aims to stimulate and improve information sharing, collaboration and learning within the operations research community by leveraging the open source and open content approach to content creation. *Information sharing* is achieved by providing a repository where all operations research problems are hosted. All information on libOR is freely available to anyone interested, hence reaching the widest audience possible. Thanks to the better availability of data sets, operations researchers can more easily apply a new algorithm to a broad selection of data sets, which will lead to more reliable and more robust results. The *collaboration* between researchers is facilitated by providing an easy to use Web-platform. Everyone in the operations research can access this website, and can also participate in the further expansion of the libOR library. The joint forces of a large community will enable more frequent updates. Operations researchers can therefore keep better track of new developments within their field, which will increase scientific advancement. *Learning* in the operations research community is enabled by the same factors as the previous two aims, with the corresponding advantages. There is however an important additional source of learning. On libOR, the results of the application of algorithms or heuristics are also published. This makes it easy for researchers to compare the performance of various approaches on the same data set, without having to resort to an extensive literature review. Moreover, the publication of results is not limited to "successful" attempts. Contrary to most traditional operations research journals, even algorithms that do not lead to a better solution will be published. This allows researchers to learn from unsuccessful approaches from other researchers, and avoid the inefficient practice of taking the same approach. This may prove to be a very important advantage, especially for researchers new to the domain.

The third contribution consists of the IT platform that was developed to support the goals described above. This platform is based on a Web-based information repository, focused on the operations research domain. The system provides a database in which research artifacts can be stored. This is an improvement compared to the existing situation, where data sets are displayed on static HTML pages. A second technical solution is the use of the XML data format to replace the existing plain-text files. The use of XML adds semantic information to the data sets, which makes them easier to read, and some errors can be more easily detected manually. Furthermore, for each problem definition, an XML

Schema will be created so that individual data sets can be automatically validated by using a validating XML parser. The use of XML also frees the operations researcher from a number of programming technicalities such as character encodings and end-of-line conventions.

During the implementation of the library, we made exclusive use of open source software. Moreover, to emphasize the open character of libOR and invite future contributions, the source code of libOR is released under an open source license. Whether operations researchers will indeed participate in the further development of libOR remain to be seen. However, we believe that we can leverage Linus' Law, and that the large user base will allow us to more quickly identify and resolve any bugs.

The success of libOR ultimately depends on the appreciation and cooperation of the operations research community. We have evaluated our initiative by using the competency rallying theory of Crowston and Scozzi [4] as adapted by Keats [14] for open content projects. Based on these recommendations for a successful open content project, we identified only one potential barrier to adoption, namely the limited knowledge of XML of the operations researcher. Based on this observation, we will take appropriate action to provide the operations research community with introductory information on XML, applied to the operations research domain. Furthermore, we will point to several tools which can support the researcher. Given the many advantages described in this paper, and the fact that we already have received many positive reactions, we expect that operations researchers are likely to invest some time in learning XML in order to realize the benefits offered by libOR.

References

1. Beasley, J.E.: OR-Library: distributing test problems by electronic mail. J. Oper. Res. Soc. **41**(11), 1069–1072 (1990)
2. Benkler, Y.: Coase's penguin, or, linux and the nature of the firm. Yale Law J. **112**(3), 369–446 (2002)
3. Christophides, N., Mingozzi, A., Toth, P. : The vehicle routing problem. In: Christophides, N., Mingozzi, A., Toth, P., Sandi, C. (eds.) Combinatorial Optimization., Wiley, Chichester (1979)
4. Crowston, K., Scozzi, B.: Open source software projects as virtual organizations: Competency rallying for software development. IEE Proc. Softw. **149**(1), 3–17 (2002)
5. Dinkelacker, J., Garg, P.K., Miller, R., Nelson, D.: Progressive open source. In: ICSE '02: Proceedings of the 24th International Conference on Software Engineering, Orlando, Florida, 19–25 May, 2002. ACM, Orlando, pp. 177–184 (2002)
6. Feller, J., Fitzgerald, B.: Understanding Open Source Software Development. Addison-Wesley, London (2002)
7. Feller, J., Fitzgerald, B., Hissam, S., Lakhani, K. (eds.): Perspectives on Free and Open Source Software. MIT Press, Cambridge (2005)
8. Fielding, R.T.: Shared leadership in the apache project. Commun. ACM **42**(4), 42–43 (1999)
9. Gendreau, M., Laporte, G., Potvin, J.Y. : Metaheuristics for the capacitated VRP. In: Toth, P., Vigo, D. (eds.) The Vehicle Routing Problem., pp. 129–154. Society for Industrial and Applied Mathematics, Philadelphia (2001)
10. Ghosh, R., Glott, R.: Free/libre and open source software: Policy support. FLOSSPOLS deliverable D3, MERIT, University of Maastricht, Maastricht (2005)
11. Hardaway, D.E.: Sharing research in the 21st century: Borrowing a page from open source software. Commun. ACM **48**(8), 125–128 (2005)
12. Katzy, B.R., Crowston, K.: A process theory of competency rallying in engineering projects. Working paper (2000)
13. Katzy, B.R., Schuh, G., Millarg, K.: Die virtuelle fabrik—produzieren in netzwerken. Technische Rundschau, pp. 30–34 (1996)
14. Keats, D.: Collaborative development of open content: A process model to unlock the potential for african universities. First Monday **8**(2) (2003)
15. Kelty, C.: Free science. In: Feller, J., Fitzgerald, B., Hissam, S., Lakhani, K. (eds.) Perspectives on Free and Open Source Software., pp. 415–430. MIT Press, Cambridge (2005)
16. Koch, S. (ed.): Free/Open Source Software Development. Idea Group, Hershey (2004)
17. Lerner, J., Tirole, J.: Some simple economics of open source. J. Ind. Econ. **50**(2), 194–234 (2002)
18. Lougee-Heimer, R.: The Common Optimization INterface for Operations Research: Promoting open-source software in the operations research community. IBM J. Res. Dev. **47**(1), 57–66 (2003)
19. Lussier, S.: New tricks: How open source changed the way my team works. IEEE Softw. **21**(1), 68–72 (2004)
20. Raymond, E.S.: The cathedral and the bazaar. In: Raymond, E.S. (ed.) The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary., pp. 19–64. O'Reilly, Sebastapol (2001)
21. Robbins, J.E: Adopting open source software engineering (OSSE) practices by adopting OSSE tools. In: Feller, J., Fitzgerald, B., Hissam, S., Lakhani, K. (eds.) Perspectives on Open Source and Free Software., pp. 245–264. MIT Press, Cambridge (2005)
22. Scacchi, W.: When is free/open source software development faster, better, and cheaper than software engineering? Working paper, Institute for Software Research, UC Irvine (2003)
23. Schweik, C.M., Grove, J.M.: Fostering open-source research via a world wide web system. Public Adm. Manage. **5**(4), 161–189 (2000)
24. Schweik, C.M., Stepanov, A., Grove, J.M.: The open research system: a web-based metadata and data repository for collaborative research. Comput. Electron. Agric. **47**(3), 221–242 (2005)
25. Sörensen, K., Ven, K., Seveaux, M., Verelst, J.: libOR – library of OR data sets. Workshop on Real-Life Applications of Metaheuristics, University of Antwerp (2003)
26. Sörensen, K., Ven, K., Seveaux, M., Verelst, J.: libOR – library of OR data sets. INFORMS Annual Meeting, Denver (2004)
27. Sowe, S.K., Stamelos, I., Samoladas, I. (eds.): Emerging Free and Open Source Software Practices.. IGI Publishing, Hershey (2007)
28. Ven, K., Sörensen, K., Verelst, J., Sevaux, M.: Stimulating collaborative development in operations research with libor. In: Scotto, M., Succi, G. (eds.) Proceedings of the First International Conference on Open Source Systems (OSS2005), Genova, Italy, 11–15 July. ECIG, Genova
29. Ven, K., Van Nuffel, D., Verelst, J. : The migration of public administrations towards open source desktop software: Recommendations from research and validation through a case study. In: Sowe, S.K., Stamelos, I., Samoladas, I. (eds.) Emerging Free and Open Source Software Practices., pp. 191–214. IGI Publishing, Hershey (2007)