Towards Creative Software Blending: Computational Infrastructure and Use Cases

Matej Martinc[1,2], Martin Žnidaršič[1], Nada Lavrač[1,3] and Senja Pollak[1]
[1] Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
[2] Jožef Stefan International Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia
[3] University of Nova Gorica, Nova Gorica, Slovenia
E-mail: matej.martinc@ijs.si, nada.lavrac@ijs.si, martin.znidarsic@ijs.si, senja.pollak@ijs.si

Keywords: computational creativity, software blending, visual programming platforms

Received: October 31, 2017

Numerous visual programming platforms support the generation, execution and reuse of constructed scientific workflows. However, there has been little effort devoted to building creative software blending systems, capable of composing novel workflows by autonomously combining individual software components or even entire workflows originally designed for solving tasks in different research fields. Based on the review of relevant computational creativity research and of contemporary web platforms for workflow construction, this paper defines the desired functionality of a software blending system. Considering the required autonomy of the system and the workflow complexity limitations, we investigate the necessary conditions for the implementation of a creative blending system within the existing visual programming platforms.

Povzetek: Številne platforme za vizualno programiranje podpirajo gradnjo, izvajanje in ponovno uporabo zgrajenih znanstvenih delotokov. Dosedanje raziskave niso posvečale pozornosti izdelavi kreativnih sistemov za spajanje programske opreme, ki bi bili sposobni avtonomnega sestavljanja posameznih programskih komponent ali celo celotnih delotokov, prvotno izdelanih za reševanje nalog na različnih znanstvenih področjih. Na podlagi pregleda raziskav s področja računalniške ustvarjalnosti in obstoječih spletnih platform za gradnjo delotokov v tem članku definiramo želeno funkcionalnost sistema za kreativno spajanje programske opreme. Upoštevaje zahteve po avtonomnosti sistema in dovoljeno kompleksnost delotokov preučimo tudi pogoje za implementacijo takega sistema v obstoječih platformah za vizualno programiranje.

1 Introduction

Creativity was defined by M. Boden [3] as "the ability to come up with ideas or artefacts that are new, surprising, and valuable". It is considered as an aspect of human intelligence, grounded in everyday abilities such as conceptual thinking, perception, memory and reflective self-criticism.

Software is usually not considered creative because it follows explicit instructions of the programmer [4]. However, writing software is considered to be a creative task. If a program could define its own instructions, this would clearly mean that the program has some level of creativity.

A subfield of artificial intelligence has recently emerged, in which one of the main goals is the creation of software that is able to model, simulate or replicate human creativity. This field, called *computational creativity*, has been defined by S. Colton and G. Wiggins [6] as "the philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative."

Note that the field of computational creativity should not be confused with the field of *creative computing*. Although these two research areas partly overlap, creative computing differs from computational creativity by generally not being considered as a subfield of artificial intelligence, since it mostly addresses the task of creative development of computing products and with how to write software that would better serve the needs of the creative community [13].

Infrastructures supporting computational creativity and the generation of creative systems are scarce, although some recent research attempts has tried to fill this gap. One of the recent developments is FloWr [4], a system for implementing creative systems as scripts over processes and manipulated visually as flowcharts. Another is the ConCreTeFlows infrastructure [27], which was developed to enable the construction, sharing and execution of computational creativity (CC) workflows, composed of software ingredients of different partners of European project ConCreTe[1]. Both of these infrastructures use different types of resources (e.g., musical, pictorial and textual inputs) in order to support the development of some typical CC task such as poetry generation, metaphor creation, generation of narratives, creation of fictional ideas and conceptual blending.

[1]`http://conceptcreationtechnology.eu`

These platforms, which enable the user to build procedures capable of producing a variety of different creative artefacts, could hardly be called creative systems, since they do not exhibit creative behavior in terms of automated workflow development. The arguably most creative system for automated workflow construction, optimization and alteration, which is implemented in the FloWr platform, requires a lot of manual user input and could only be called creative with some major reservations.

To fill the identified gap, this paper addresses the task of developing an infrastructure capable of autonomously composing novel scientific workflows by creatively combining individual software components or even entire workflows originally designed for specific tasks in different research fields. We consider the process of autonomous workflow composition—which we name *creative software blending* in this paper—to be an important first step towards a long term goal of creating software that could write code directly. The proposed system would be able to bridge different scientific fields by combining methods from specific fields into novel interdisciplinary workflows. It would ideally also be capable of automated interdisciplinary research by autonomously discovering novel scientific procedures.

This paper presents the design principles underlying a creative system described above. Section 2 introduces the research topic and presents the infrastructures suitable for the implementation of a creative software blending system. Section 3 motivates this research by presenting two existing hand-blended workflows. Section 4 presents the related software blending and computational creativity research, followed by an outline of the desired system functionality, investigating the necessary conditions for the implementation of a creative system for autonomous creative workflow generation. The paper concludes by presenting plans for future work.

2 Research background and infrastructures

As background to our creative software blending research, this section first outlines some creativity support tools, followed by a brief description of a selection of easy-to-use workflow management systems that allow the user to compose complex computational pipelines in a modular visual programming manner.

2.1 Creative software

As Colton's and Wiggins' definition of computational creativity [6] is hardly operational for measuring creativity of a program, G. Ritchie [23] proposed some empirical criteria for attributing creativity to a computer program. The main idea is to use empirically observable and comparable factors, such as the properties of the generated output of the creative system, when trying to assess the creativity of a system. These observable factors can be judged by two quantifiable and essential criteria:

Novelty of an output determines to what extent is the produced item dissimilar to existing examples of its genre.

Quality of an output determines to what extent is the produced item a high quality example of its genre.

Using these criteria, we can say that the system for creative software blending is creative if it outputs novel and high quality scientific workflows.

Another relevant question is what types of creative behaviors exist and how can they be computationally modeled. Boden [3] distinguishes three basic types of creativity:

Combinational creativity involves making unfamiliar combinations of familiar ideas.

Exploratory creativity involves exploration of a conceptual space, which is characterized as a structured style of thought, and coming up with a new idea or artefact within that thinking style.

Transformational creativity refers to the modification of the conceptual space so that new kinds of ideas and artefacts can be generated.

Combinational creativity is the easiest one to be modeled on a computer. However, created combinations should be meaningful and interesting, which usually requires a solid background knowledge and the ability to form and evaluate relations of many different types. Several programs exist that can explore a given space and invent new artefacts with a certain style, for example, a program for automatic music generation [19] or a program for generating game designs [7]. Some programs can even transform their conceptual space by altering their own rules; for example, evolutionary algorithms can make random changes in their current rules and by this evolve new structures.

Another important distinction made by Boden [3] is a distinction between *psychological creativity* (P-creativity) and *historical creativity* (H-creativity). P-creativity relates to creation of surprising, valuable ideas and artefacts that are new to the person who comes up with it. However, if an artefact or idea has arisen for the first time in human history and (so far as we know) nobody else has had it before, then we are talking about H-creativity. We anticipate that if the targeted creative software blending system is to be an active participant in scientific discovery or artefact creation, it should ideally be H-creative, although even a P-creative system can play a very useful supporting role in scientific research and its development is therefore a worthy research goal.

2.2 Infrastructures

A system for creative software blending would best be implemented inside an already existing infrastructure enabling interdisciplinary and creative scientific workflow composition. In this section we present the ClowdFlows and ConCreTeFlows platforms that host the two motivational use cases, but other platforms, such as FloWr [4], Rapid Miner [18], KNIME [2], ORANGE [8] are also worth exploring as potential infrastructures for creative software blending.

ClowdFlows [16] is a cloud-based web application[2] for composition, execution and sharing of interactive data mining workflows. It has a web based user interface for building workflows, runs in all major browsers and requires no installation. It contains a large set of workflow components called *widgets*, which can be connected in a specific meaningful order to create a *workflow*. ClowdFlows enables visual programming and has a graphical user interface which consists of a widget repository and a workflow canvas.

ConCreTeFlows [27] is a platform[3] built on top of the ClowdFlows infrastructure. It is specialized in computational creativity tasks, including conceptual blending based on textual or visual input or text generation tasks, such as poetry generation.

The specialization of ConCreTeFlows in computational (and especially text-based) creativity, as well as a smaller number of implemented widgets, makes it less appropriate for the implementation of the proposed system for creative software blending, but it is appropriate to showcase the creative blending process. On the other hand, ClowdFlows is not specialized in a single specific research field and contains widgets from the fields of text mining, machine learning and NLP, which makes it appropriate for the implementation of a creative software blending system since combining tools from different research fields would most likely increase the chance of the system to be H-creative. As a basis of automated software composition, ClowdFlows already includes a—somewhat loosely defined—ontology of its components (named widgets), which should be enhanced and elaborated in further work, to enable ClowdFlows to actually become a useful infrastructure for software blending.

3 Motivational use cases

This section presents two hand-crafted motivational workflows, which illustrate the usefulness of blending software from different scientific fields in order to develop new innovative scientific methods. In this sense, they represent the type of workflows that a system for creative software blending would be capable to produce.

3.1 Wordification use case: Blending data mining and text mining in ClowdFlows

Propositionalization [15] is an approach to inductive logic programming (ILP) and relational data mining (RDM), which offers a way to transform a relational database into a propositional single-table format. Consequently, learning with propositionalization techniques is divided into two self-contained phases: (1) transformation of relational data into a single-table format and (2) selecting and applying a propositional learner to the transformed data set. As an advantage, propositionalization is not limited to specific data mining tasks such as classification, which is usually the case with ILP and RDM methods that directly induce predictive models from relational data. This section motivates creative software blending by outlining the Wordification workflow [22], implemented in ClowdFlows, which performs propositionalization by combining data mining and text mining techniques.

In the Wordification workflow, shown in Figure 1, given a MySQL relational database as input, the user selects the target table from the initial relational database, which will later represent the main table in the Wordification component of the workflow. The user is able to discretize each of the tables using one of the available discretization techniques. These discretized tables are used by the Wordification widget, where the transformation from the relational tables to a 'corpus of documents' is performed.

Several elements of blending data mining and text mining techniques are incorporated in the Wordification: i.e. transforming attribute values into bags of word-like items, using TF-IDF weighting of items, and the possibility of using n-grams of items where n-gram construction is performed by taking every combination of length n of items from the set of all items corresponding to the given individual. Nevertheless, the element of the workflow that most clearly illustrates the software blending potential is the inclusion of a *word cloud* visualization (an approach developed in text mining research), together with decision tree construction and visualization (an approach developed in data mining research).

3.2 Conceptual blending use case: computational creativity in ConCreTeFlows

The elements of the *conceptual blending* theory [12], described in more detail in Section 4, are an inspiration to many algorithms and methodologies in the field of computational creativity. In brief, according to this theory, two different concepts for which we can define (find) a similarity, can be blended into a new concept in the context of knowledge that is necessary to represent and generalize the two concepts.

[2]Available at `http://clowdflows.org`
[3]Available at `http://concreteflows.ijs.si`

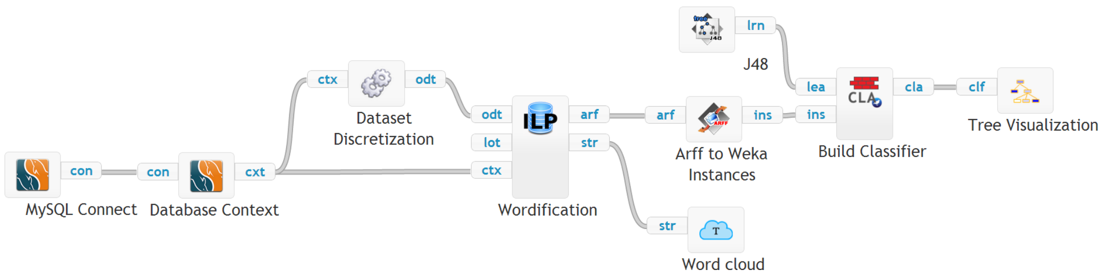

Figure 1: Clowdflows Wordification workflow with additional analyses after the wordification process, available at `http://clowdflows.org/workflow/1455/`.

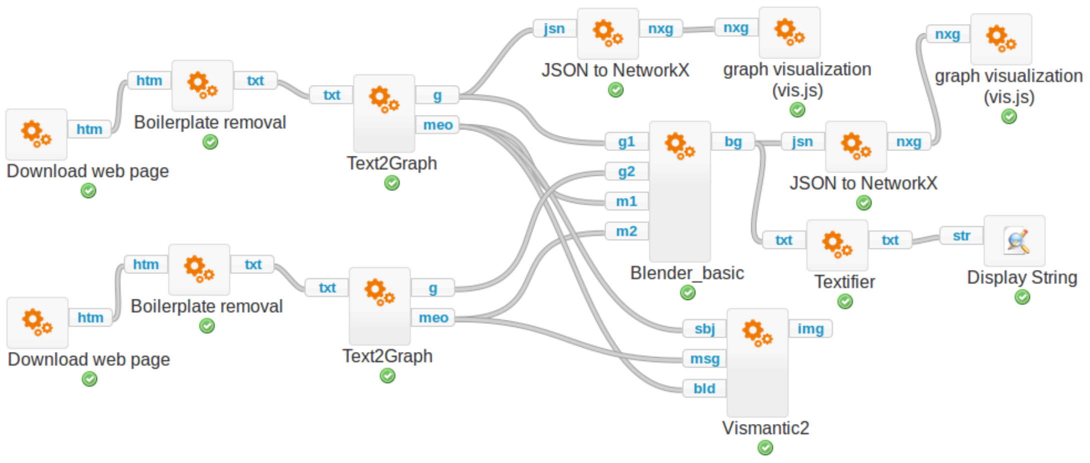

Figure 2: Workflow implementation of multimodal blending in ConCreTeFlows, available at: `http://concreteflows.ijs.si/workflow/137/`.

Let us present a conceptual blending CC workflow [27], implemented in the ConCreTeFlows platform by different partners of the ConCreTe project. Its process components are implemented either as internal functions, wrapped standalone programs or as Web services. The publicly available workflow, presented in Figure 2, can be executed, changed and extended with additional functionality.

The workflow presents conceptual blending by constructing conceptual graphs from textual input and representing the results (blends) as graphs, natural language descriptions and visual representations. Two textual inputs are transformed into conceptual graphs by a series of widgets: the *Download web page* for obtaining the Web page source from a given URL (In the example, these are the Wikipedia pages for two animals: hamster and zebra.), *Boilerplate removal* and *Text2Graph* transforming the textual content into conceptual graphs (output *g*). The outputs of Text2Graph widgets enter *Blender_basic*, which blends the two graphs together and outputs a combined blended graph (output *bg*). This one gets served to the *Textifier widget*, which produces a textual description of the blend. Its output is presented by a standard *Display String* widget. The two main entities from Text2Graph widgets enter also the *Vismantic2* visual blending widget [28], which either changes the texture of one input space to the texture of the other (see Figure 3a), or puts one in the usual surroundings of the other. (Figure 3b). Its outcome is shown in an output similar to the ones shown in Figure 3.

4 Towards design principles for creative software blending

There are two major paradigms in artificial intelligence research: problem solving and artefact generation [5]. While the problem solving paradigm deals with a series of problems that needs to be solved, in the artefact generation paradigm the task is to generate a series of valuable artefacts. This study is more related to the latter and the artefacts of our interest are functional workflows.

A creative software blending system should be able to build new workflows composed of software components from different fields, leading to novel ways of software composition for computational purposes that were not expected in advance. Such blending of software would best be implemented in an existing infrastructure for interdisciplinary scientific research with already implemented components for specific and well defined tasks.

As shown in Section 2.2, much effort in the fields of data mining and NLP has already been devoted to the development of infrastructures that provide support for easier and quicker experimentation. One of the biggest challen-

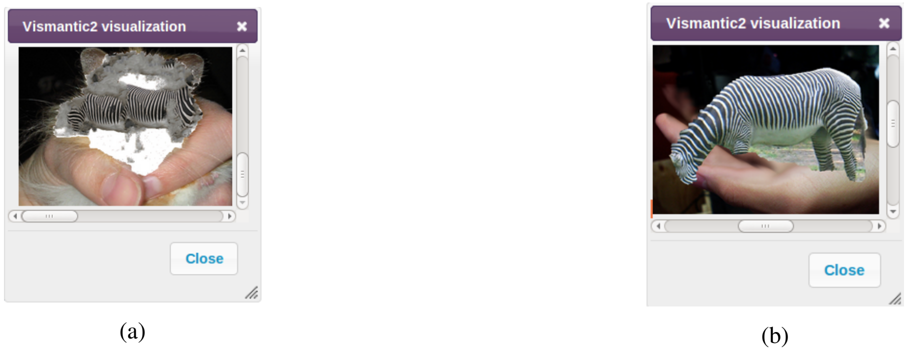

Figure 3: Two outputs of the Vismantic2 widget for the example of blending the concepts of *hamster* and *zebra*: left is a result of exchanging hamster's texture with zebra's and the right is an example of exchanging zebra's with a hamster's common visual context.

ges in implementation and use of these infrastructures has been the integration of different components into functional workflows. Combining different tools and technologies in a common infrastructure is a difficult task because of software incompatibility and inappropriately defined ontologies.

4.1 Related software blending research

To design an appropriate creative software blending system one should consider three fields of study. First, one has to reflect upon the concept of creativity and how to build software that exhibits creative behavior (see the related research in Section 2.1). Next, one has to be aware of strengths and limitations of the existing infrastructures that could be used as a platform for the implementation of our system (see the related infrastructures in Section 2.2). Finally, one has to become aware of potentially existing implemented approaches for software blending, surveyed below.

While the FloWr framework [4] is conceptually very similar to the two infrastructures described in Section 2.2, it is currently the only one with a specifically defined aim of being able to automatically optimize, alter and ultimately generate novel workflows presented as flowcharts. This automatic workflow generation via the combination of code modules means that FloWr has the potential to innovate at the process level and the manifested long-term goal is a software system that can write program code for itself [4]. Although the platform currently does not support fully functioning software blending, some preliminary experiments to automatically alter, optimize and generate flowcharts have been conducted.

One of the FloWr experiments dealt with an automatic construction of a system for producing poetic couplets from scratch. In order to reduce the number of possible combinations of different workflow components, only a subset of all the available components were manually selected for blending in the experiment. Possible options for the input parameters were manually reduced and the number of components in the generated workflow was limited to 3 to 5. Despite these limitations, at the end there were still over 261 million variable definition combinations. For this reason the brute-force approach of testing all combinations was intractable, so a depth-first search for all possible workflows was implemented in a way that just one node combination and one parameter setting were randomly selected from a set of allowed combinations. The compatibility of sequential components and some other restrictions were taken into account, which reduced the number of possible workflow candidates. The algorithm was run 200 times resulting in 200 workflows. A manual evaluation showed that 18.5% of workflows worked successfully and produced poetic couplets.The conducted experiment required a lot of human intervention in order to be successful and the evaluation of produced artefacts was done by humans. Because of this we can question the creativity of the proposed software blending approach since a software should — at least in our opinion — have the capacity to evaluate its own performance in order to be called creative.

While FloWr belongs to CC research, several attempts have been made to develop support systems also in the field of knowledge discovery. These systems are to some extent related to our research, since they either support the users workflow composition by recommending the new components that could be attached to an existing workflow, or by generating entire workflows according to user requirements.

Zakova et al. [26] proposed a semi-automatic system for workflow generation that is based on a background knowledge ontology in which all workflow components are described together with their inputs, outputs and pre-/post conditions. The system uses a planning algorithm and returns just one optimal workflow with the smallest number of processing steps. Given that alternative workflows are not generated, this is not in accordance with a desired system for creative software blending. Complexity limitations are another problem, which is common to all the systems that use planning approaches for workflow generation.

The IDEA system by Bernstein et al. [1] is based on an ontology of data mining components that guides the workflow composition and contains heuristics for the rankings of different alternatives. The system does not enable fully

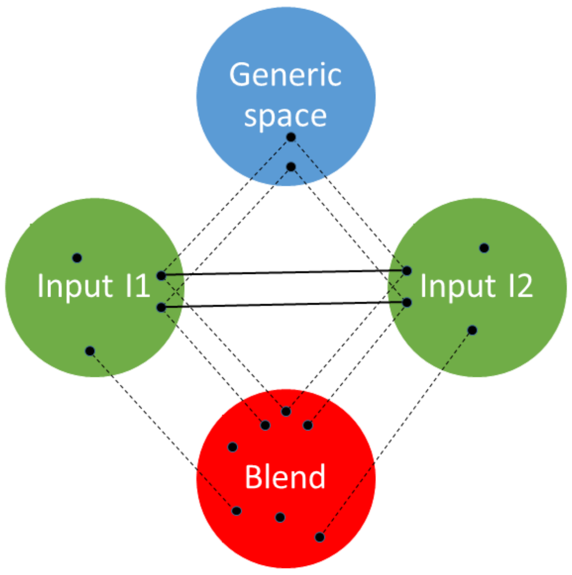

Figure 4: The conceptual blending network [11].

automatic workflow generation but was implemented as a support system for the user who decides on the weights to determine the trade-off between different performance criteria (e.g., speed, accuracy, comprehensibility). IDEA is limited to proposing fairly simple workflows.

Kietz et al. [14] proposed a KDD support system that uses a data mining ontology. The ontology contains information about the objects manipulated, the meta data, the operators (i.e. components containing algorithms for specific tasks) and a description of the goal, which is a formalization of the user desired output. The system takes a goal description as an input and returns a workflow together with all the evaluation and reporting needed to let the user assess if it fulfills the user defined success criterion. The system, implemented in the RapidMiner platform, is not fully autonomous, as it was designed as a support system for the user.

4.2 Design principles

As the above survey shows, no adequate solution for the autonomous creative software blending currently exists. To build such a system, first, an ontology with well-defined rules and relations needs to be created, in order to enable combining software components in a meaningful way. Next, a system for creative blending of software components would be created, enabling automated combination of components in functional workflows.

Computational creativity, which is still in early phrases of its development [25], provides some methodological apparatus and inspiration for designing the guiding principles for a creative software blending system. One of the very productive fields of research in computational creativity is the *conceptual blending* (CB) theory [12], which inspired many algorithms, methodologies and discussions in the field (e.g., [24, 21, 17]).

CB is a basic mental operation that leads to new meaning, global insight, and conceptual compressions useful for memory and manipulation of otherwise diffuse ranges of meaning [9]. A key element is the *mental space*, a partial and temporary structure of knowledge built for the purpose of local understanding and action [10].

To describe the CB process, the theory [12] makes use of a network of four mental spaces (see Figure 4). In blending, structure from two input mental spaces (*Input* I_1, *Input* I_2), is projected to a new space, the blend. A partial mapping between elements of input spaces—that are perceived as similar or analogous in some respect—is performed. The third mental space, called *generic space*, encapsulates the conceptual structure shared by the input spaces, generalizing and possibly enriching them. This space provides guidance to the next step, where elements from each of the input spaces are selectively projected into the *blend*, i.e. the new blended mental space. Emergent structure arises in the blend that is not copied there directly from any input.

The conceptual blending model is not directly transferable from the human cognition to the blending of software. However, the methodology, together with the optimality principles [11] that optimize the blending process—which were already addressed also in computational models [20]—should be considered when implementing the software blending algorithm and the workflow ontology. For example, in software blending the two inputs would not represent concepts but rather two workflows from two different scientific domains. The "generic space" could then be adapted to software blending in order for the blending system to find all the compatible widgets from two different input workflow domains. Finally, the blend would be a newly produced workflow containing new emergent structures not copied from original workflows. The optimality principles, such as the relevance principle (which dictates that all elements in the blend should be relevant) and integration principle (which states that the final blend should be perceived as an enclosed unit) should be kept in mind when designing the ontology.

Another important aspect to be considered in the implementation of the system is its creative part. In order for the system to be recognized as creative, its produced artefacts should be novel and of good quality [23] and the human interference in the production and evaluation of these artefacts should be minimal. Three criteria are proposed for attributing creative autonomy to a system [25]:

Autonomous evaluation The system should be able to evaluate new creations autonomously and possess its own "opinion" on which creations are better than others.

Autonomous change The system should be able to change its evaluation function without explicit directions.

Non-Randomness (Aleatoricism) Random behavior is not creative, so evaluation and change should not be completely random, although some randomness can be involved.

In order to satisfy these criteria and since most of the aforementioned platforms contain a large set of manually built workflows that could be used as a training set, we propose a combination of an evolutionary algorithm and a classification model induction. An evolutionary algorithm would operate directly on representations of workflows and generate new workflow candidates, according to the constraints defined by ontology rules. These constraints would enforce a minimum quality for the produced workflows (corresponding to the criterion of quality [23], which is, as explained earlier, one of the guiding principles in the construction of creative artefacts).

The initial population of the evolutionary algorithm would consist of manually built workflows that would be "blended" into new workflow candidates with the help of mutation and crossover. The fitness function used for evaluating the fitness of the generated workflow candidates would contain following elements:

A binary classification model trained on the features extracted from successful and unsuccessful workflows would serve as an additional workflow quality check.

A similarity function for determining the similarity between a generated workflow candidate and existing workflow would be used for evaluating the novelty of the candidate.

In this way the system would be able to generate new—possibly creative—workflows and even propose changes in the existing rules for workflow generation, which would make this system capable of transformational creativity according to [3].

5 Conclusions

In this study we elaborate the initial design principles of a system for automatic workflow generation that would be capable of autonomous composition of novel workflows from existing software components. We have presented two workflows with human-designed blending, implemented in the ClowdFlows and ConCreTeFlows platforms for online workflow composition. The first workflow clearly illustrated the potential for the composition of computational creativity solutions. The second use case presents several computational creativity software components that were combined in a collaborative effort to implement an interesting conceptual blending solution, resulting in conceptual, visual and textual blends. The benefits of a unifying workflow for blending are twofold: the user can get blends of various kinds through the same user interface and the components can affect one another to produce a more coherent and orchestrated set of multimodal blending results. The presented prototype solution is fully operational and serves as a proof of concept that such an approach to multimodal conceptual blending is possible.

On the other hand, the sketched evolutionary algorithm approach to blending workflows and workflow components shows, that the theory of conceptual blending can be transferred to the problem of creative software blending. We also demonstrated that the system will be capable of self evaluation by using the empirical criteria of novelty and quality in the fitness function.

In our future work we will first design an ontology capable of supporting the planned widget recommender system. We also plan to integrate a larger number of widgets and workflows in the presented platforms. Moreover, we will undertake the challenging task of the implementation. We realize that creation of software that can innovate at a process level is a very demanding task and we can expect many challenges during this phase. Anyhow, we do believe that the effort will be fruitful and bring us closer to the long-term goal of creating software that could write novel and valuable code directly.

Acknowledgments

We acknowledge the support of the Slovenian Research Agency through research programme Knowledge Technologies (grant number P2-0103), and project ClowdFlows Data and Text Analytics Marketplace on the Web (CF-Web), which has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 754549. We would like to thank Pedro Martins and Amilcar Cardoso for numerous discussions on the topic of conceptual blending.

References

[1] Bernstein, A., Provost, F., Hill, S.: Toward intelligent assistance for a data mining process: An ontology-based approach for cost-sensitive classification. IEEE Transactions on Knowledge and Data Engineering 17(4), 503–518 (2005)

[2] Berthold, M.R., Cebron, N., Dill, F., Gabriel, T.R., Kötter, T., Meinl, T., Ohl, P., Thiel, K., Wiswedel, B.: Knime–the Konstanz Information Miner: Version 2.0 and beyond. ACM SIGKDD Explorations Newsletter 11(1), 26–31 (2009)

[3] Boden, M.A.: Creativity in a nutshell. Think 5.15, 83–96 (2007)

[4] Charnley, J., Colton, S., Llano, M.T.: The FloWr framework: Automated flowchart construction, optimisation and alteration for creative systems. In: Proc. of the Fifth International Conference on Computational Creativity. pp. 315–323 (2014)

[5] Colton, S., Ramezani, R., Llano, M.: The hr3 discovery system: Design decisions and implementation

details. In: Proc. of the AISB Symposium on Computational Scientific Discovery (2014)

[6] Colton, S., Wiggins, G.: Computational creativity: The final frontier? In: Proc. of the 20th European Conference on Artificial Intelligence. pp. 21–26 (2012)

[7] Cook, M., Colton, S.: Multi-faceted evolution of simple arcade games. IEEE Conference on Computational Intelligence and Games (CIG) pp. 289–296 (2011)

[8] Demšar, J., Zupan, B., Leban, G., Curk, T.: Orange: From experimental machine learning to interactive data mining. In: European Conference on Principles of Data Mining and Knowledge Discovery. pp. 537–539. Springer (2004)

[9] Fauconnier, G., Turner, M.: Conceptual blending, form and meaning. Recherches en communication 19(19), 57–86 (2003)

[10] Fauconnier, G.: Mental Spaces: Aspects of Meaning Construction in Natural Language. Cambridge University Press (1994)

[11] Fauconnier, G., Turner, M.: Conceptual integration networks. Cognitive Science 22(2), 133–187 (1998)

[12] Fauconnier, G., Turner, M.: The way we think: Conceptual blending and the mind's hidden complexities. Basic Books (2002)

[13] Hugill, A., Yang, H.: The creative turn: new challenges for computing. International Journal of Creative Computing 1(1), 4–19 (2013)

[14] Kietz, J., Serban, F., Bernstein, A., Fischer, S.: Towards cooperative planning of data mining workflows. In: Proc. of the Third Generation Data Mining Workshop at ECML/PKDD-2009. pp. 1–12 (2009)

[15] Kramer, S., Lavrač, N., Flach, P.A.: Propositionalization approaches to relational data mining. In: Džeroski, S., Lavrač, N. (eds.) Relational Data Mining, pp. 262–292. Springer (2001)

[16] Kranjc, J., Podpečan, V., Lavrač, N.: ClowdFlows: A cloud based scientific workflow platform. In: Proc. of ECML/PKDD (2). pp. 816–819. Springer (2012)

[17] Martins, P., Pollak, S., Urbancic, T., Cardoso, A.: Optimality principles in computational approaches to conceptual blending: Do we need them (at) all? In: Proceedings of the Seventh International Conference on Computational Creativity, UPMC, Paris, France, June 27 - July 1, 2016. pp. 346–353 (2016)

[18] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: Rapid prototyping for complex data mining tasks. In: Proc. of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 935–940. ACM (2006)

[19] Monteith, K., Martinez, T., Ventura, D.: Automatic generation of music for inducing emotive response. In: Proc. of the International Conference on Computational Creativity. pp. 140–149 (2010)

[20] Pereira, F.C., Cardoso, A.: Optimality principles for conceptual blending: A first computational approach. AISB Journal 1, 4 (2003)

[21] Pereira, F.C.: Creativity and AI: A Conceptual Blending approach. Ph.D. thesis, Dept. Engenharia Informática da FCTUC, Universidade de Coimbra, Portugal (2005)

[22] Perovšek, M., Vavpetič, A., Cestnik, B., Lavrač, N.: A wordification approach to relational data mining. In: Proc. of the International Conference on Discovery Science. pp. 141–154. Springer (2013)

[23] Ritchie, G.: Some empirical criteria for attributing creativity to a computer program. Minds and Machines 17.1, 67–99 (2007)

[24] Schorlemmer, M., Smaill, A., Kühnberger, K.U., Kutz, O., Colton, S., Cambouropoulos, E., Pease, A.: COINVENT: Towards a computational concept invention theory. In: Proc. of the 5th Int. Conference on Computational Creativity. pp. 288–296 (2014)

[25] Toivonen, H., Gross, O.: Data mining and machine learning in computational creativity. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 5.6, 265–275 (2015)

[26] Žakova, M., Kremen, P., Železnỳ, F., Lavrač, N.: Planning to learn with a knowledge discovery ontology. In: Proc. Planning to Learn Workshop (PlanLearn 2008). vol. 951 (2008)

[27] Žnidaršič, M., Hervás, R., Alves, A.O., Oliveira, H.G., Xiao, P., Linkola, S., Toivonen, H., Kranjc, J., Lavrač, N.: Computational creativity infrastructure for online software composition: A conceptual blending use case. In: Proc. of the 7th International Conference on Computational Creativity (2016)

[28] Xiao, P., Linkola, S.: Vismantic: Meaning-making with images. In: Proceedings of the Sixth International Conference on Computational Creativity. pp. 158–165. ICCC2015 (Jun 2015)