Sustainable Design and Manufacturing 2017: Selected Papers on Sustainable Design and Manufacturing
SDM 2017. Smart Innovation, Systems and Technologies, volume 68, pp. 172-184
ISBN:978-3-319-57077-8; DOI:10.1007/978-3-319-57078-5_17

A tool to promote sustainability in casting processes: development highlights

Emanuele Pagone, Mark Jolly, Konstantinos Salonitis

Manufacturing Department, Cranfield University, Cranfield, MK43 0AL, United Kingdom
{e.pagone,m.r.jolly,k.salonitis}@cranfield.ac.uk

Abstract. The validity of traditional manufacturing decision variables (i.e. cost, quality, flexibility and time) is questioned by some important challenges of our time: the scarcity of natural resources and environmental pollution. Increasing energy cost to extract and process natural resources, alongside regulatory pressures against pollution, pushes very mature and competitive processes like casting towards a holistic approach where sustainability contributes to strategic decisions together with the mentioned traditional manufacturing variables. As a contribution to this industrial necessity, a modular tool able to analyse material and energy flows in casting processes is under development. In particular, the ability to represent automatically Sankey diagrams of the flows recently implemented is described and validated.

Keywords: Sustainable manufacturing systems; Energy efficiency; Casting.

1 Introduction

Traditional manufacturing decision variables comprise cost, quality, flexibility and time [1]. However, in recent times scarcity of resources and pollution problems (e.g. the Paris UN Climate Change Conference agreement in December 2015) are imposing a revision or integration of these standards that are not able to take these aspect into consideration.

Moreover, the increasing competition derived by the globalised economy, depicts a challenging scenario for the whole manufacturing industry and, in particular, for a mature technology like metal casting.

adfa, p. 1, 2011.

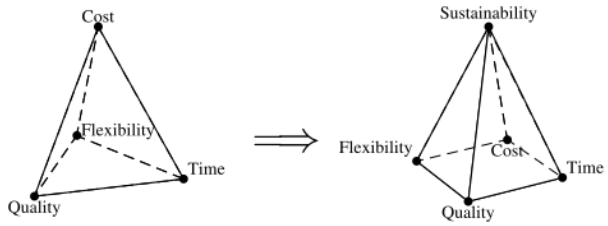

Fig. 1. Shift of decision-making attributes in modern manufacturing to include sustainability.

One promising solution to these challenges is to implement sustainability alongside traditional manufacturing decision variables [1] (Figure 1). Sustainability has been defined rather broadly as meeting the present needs of mankind without hindering the same opportunity to future generations [2]. In more pragmatic terms, a "triple bottom line" is generally accepted where environmental, economic and societal aspects are considered together [3].

The tool presented in this paper has been designed to support this change although the current effort is more focussed on energy resilience and environmental aspects. However, its modular architecture allows future integrations to potentially encompass the entire sustainability spectrum.

2 Aims and Rationale

To discuss the challenges presented in the introduction, Cranfield University, with the collaboration of the UK Cast Metals Federation (CMF), organised a workshop that was attended by several stake-holders of the sector (i.e. foundries, suppliers, consultants and academics) [4,5]. Among the key outcomes of the event, it was highlighted the need for a tool able to rapidly analyse foundry measurements. The computer program presented in this work implements this tool in a user friendly fashion. In particular, the focus is on the effective visualisation of material and energy flows encompassing the entire chain of processes from charge to waste. The program is designed to be used as a stand-alone tool or to be integrated with existing manufacturing tools (see Section 5). Its main goal is to support decision-making, offering a range of capabilities such as explore improvements, identify synergies or opportunities for energy scavenging, benchmark practices or train personnel to a correct behaviour. These features intend to promote a more energy resilient casting practice with the long-term potential to incorporate also the three areas of sustainability. Moreover, attention has been devoted to allow flexibility in selecting the type and level of detail of the input data required to adapt to different types of foundries (different processes) and different data sets available (very detailed versus broad information).

3 Energy and material flows

As briefly mentioned, the energy intensive nature of foundries make them particularly sensitive to the cost of energy because it affects significantly their competitiveness. The on-going depletion of natural resources (including fossil fuels) clearly appears one important motivation to minimise energy inputs in foundries.

However, efforts aimed at the direct reduction of the energy required by casting processes must be supported by equally important endeavours related to material flows. In fact, a significant amount of energy is spent to melt metal that does not eventually remain part of the finished product. Relevant examples are the removal of the gating and oxidised or deteriorated scrap metal. One effective metric to represent the overall performance of the process with respect to material flows of the cast metal is the Operational Material Efficiency (OME), i.e. the ratio of the amount of shipped casting over the metal melt measured in a representative amount of time [6].

However, also ancillary material flows can impact negatively (and sometimes significantly) on the energetic performance of the plant. Examples of this type of material losses are the release in the environment of flue gasses obtained by the combustion of fossil fuels and the replacement or reclamation of sand in casting processes that make use of it.

Valid strategies applicable to the improvement of casting practices include audits and lean philosophy tools [7]. However, both show shortcomings for their implementation. Audits usually suggest improvements in the equipment [8,9] that require capital investments often not viable for Small and Medium Enterprises (SMEs) like foundries [10,11,12]. Lean philosophy tools have the advantage of not requiring large investments but are usually less implemented in businesses like foundries where large stocks of raw materials are present [7].

Sankey diagrams can be an effective tool to better understand the process (as exemplified by Figure 2) and aid decision-making [13]. Hence, it appears appropriate the decision to implement the ability to automatically generate Sankey diagrams in the computer program presented.

Previous research efforts have been devoted to the modelling and visualisation of processes in the form of Sankey diagrams. Viere *et al*. implemented a modelling methodology based on Petri networks with the inclusion of economic and environmental aspects in the energy and material flows [14]. Other examples are more focussed on the visualisation in the energy sector (e.g. flows related to primary and secondary fuels) [15]. Interestingly, Sankey diagrams have been considered unsuitable for real-time visualisation because "inherently lumped over a certain time period" and they "lack dynamic" [16]. However, the ability of the computer program presented in this work to automatically generate diagrams allows its encapsulation in real-time manufacturing systems (as explained in Section 5). Another key aspect of originality of the tool is its open source nature in contrast with the commercial nature of mature implementations present in the literature (e.g. Umberto, e!Sankey pro). Finally, it should be noted that Sankey diagrams are only one visualisation option among others that can be flexibly implemented and selected in the program.

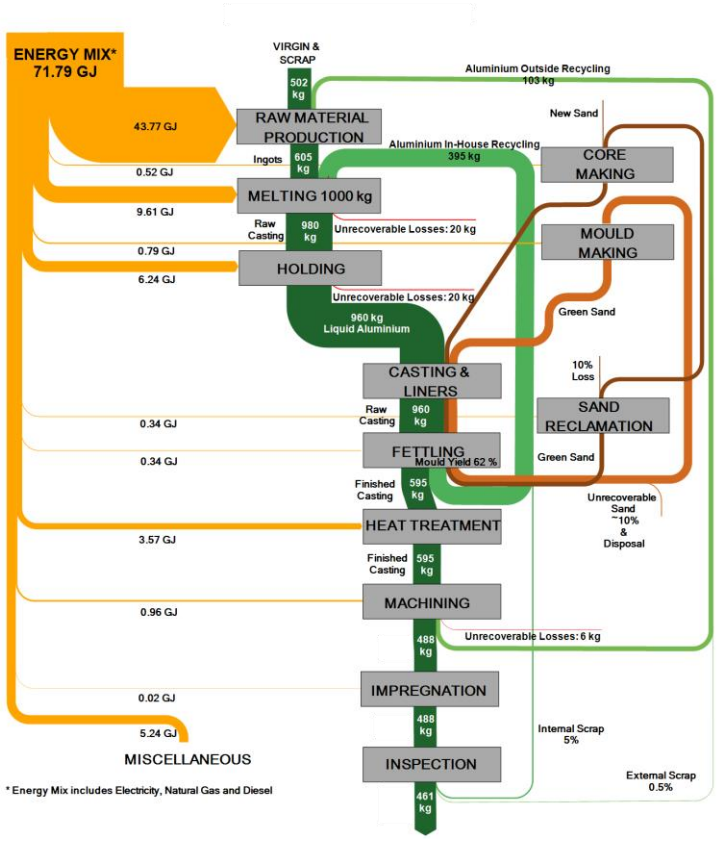

Fig. 2. Example of material and energy flows in a foundry represented by a Sankey diagram.

4 Methods

A general overview of the current state of development of the computer program is initially provided, whereas a more detailed analysis of the Sankey diagram module is provided in a separated section.

4.1 Computer program overview

A broad representation of the tool work-flow shows three main stages (Figure 3).

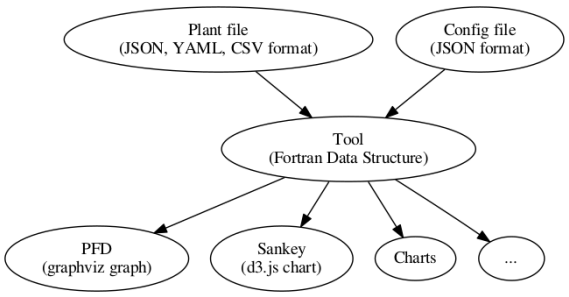

Fig. 3. General work-flow of the presented data visualisation tool.

Firstly, textual input files are prepared in specific formats to describe the foundry specifications (“Plant file”) and provide configuration settings (“Config file”). It is mandatory for the user to provide information about the plant, whereas the inclusion of non-default configuration specifications is entirely optional. Work is ongoing to create a Graphical User Interface (GUI) that will collect the information from the user in a more user-friendly manner and will generate the relevant files automatically.

Secondly, after having parsed the input files, a data structure is generated (specifically, a derived data type according to the Fortran computer language parlance). Finally, an automatic conversion of this information to a graphical representation is executed. In a modular fashion, different graphical outputs are available and currently the program can generate Process Flow Diagrams (PFDs) by means of a graphviz [17] directed graph and Sankey diagrams using the javascript library d3.js [18,19].

The data structure that contains the information about the foundry comprises an ordered series of process phases ("Components" in the "Plant file"). Each phase may include a variable number of inward or outward flows (including none) that can be categorised as of material or energy type. At least one process phase must be entered, as well as the name of the first phase of the entire process.

Foundry data can be collected with great flexibility adapting to the specific measurement system already available because there is no strong requirement on the level of accuracy required. Thus, either aggregated or instantaneous data can be provided and a sanity check based on the consistency of the unit of measures is performed by the program. However, the number of checks and restrictions imposed to the input data (concerning, for example, the sequence of different phases) has been kept to a minimum to avoid hindering flexibility. Potential gaps in the measurements of the inputs are not currently addressed directly by the program: it is the user that has to decide to change the level of accuracy that describes the plant or calculate or assume the missing datum or data. In fact, the tool is conceived primarily as an effective visualisation program whereas the simulation capabilities are kept to a minimum (basic conservation laws). However, it is not excluded that future development will also implement more extended simulation capabilities.

Moreover, the standardised format of the textual input files provides the advantage to link with relatively small effort the visualisation tool to external programs or databases (see Section 5).

The general design of the tool implements object-oriented concepts. Among the advantages of this approach, a notable example is the implementation of the routine to perform an action on all the elements of the plant, exploring the entire data structure automatically (e.g. calculate values at each "component" according to conservation laws). Such routine is a polymorphic object that applies the same algorithm (implemented only once) for every post-processing task required (more details will be provided in Section 4.2).

In a previous paper, more implementation details of the overall program and the Process Flow Diagram (PFD) module (including its validation) were presented [20].

4.2 Sankey diagram module

Compared to the general work-flow of Figure 3, a more detailed look at the process to convert the internal data structure into a Sankey diagram (an object generated by the open source javascript library d3.js), reveals a few intermediate steps (Figure 4).

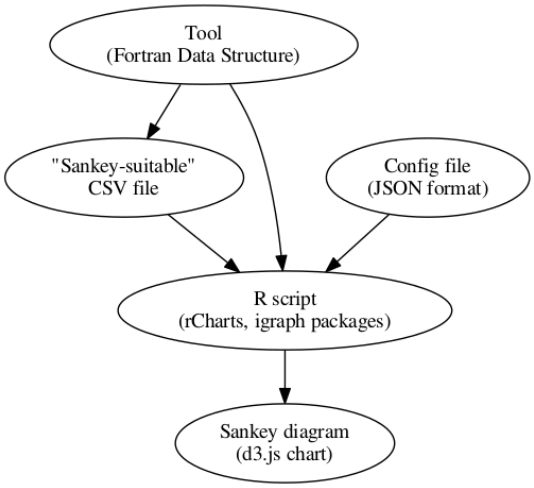

Fig. 4. Sankey diagram generation work-flow of the proposed tool.

Initially, the derived data type is converted into a CSV format file, filtering and structuring the information to be subsequently processed by a script in the R programming language [21], automatically generated in parallel. The R script uses the packages rCharts [22] and igraph [23] to convert the mentioned CSV file into the final Sankey diagram object using the specifications present in the configuration file.

The original implementation of the Sankey diagram d3.js plug-in [24] did not support the representation of "cycles", i.e. flows of energy or material that are fed back to a previous step of the overall process (e.g. re-melting scrap metal after fettling). The name "cycles", adopted from the terminology of graph theory, is also used in the implementation to name the relevant derived data types and type bound procedures. A modified version of the Sankey diagram d3.js plug-in [25] that allows the presence of "cycles" has been interfaced to the presented work-flow.

A more detailed description of the implementation is provided analysing a minimal UML class diagram that completely describes the Sankey module (Figure 5). The UML chart (obtained using the ForUML tool [26]) comprises the involved derived data types, type bound procedures and their connections. The naming of type bound procedures follows the well-established convention of "set", "get" and "predicate" methods [27].

In the lower part of Figure 5 (from right to left) are shown the essential derived data types that implement the model of the manufacturing plant. At the highest level, type `plant_t` contains the information about the entire plant and encapsulates a number of `comp_t` types, i.e. the phases that describe the overall process (e.g. melting, pouring, machining). Type `comp_t` encapsulates a variable number of `flow_t` elements that represent the inward or outward, material or energy flows of that specific phase of the process. In order to model cycles, a `cycle_t` derived data type is encapsulated in the `flow_t` type. A more accurate description of these data types and their methods is presented in a previous paper [20].

In the upper part of Figure 5, the data types and associated methods of the R script file (`R_script_t`), the CSV file (`csv_t`) and the Sankey diagram (`sankey_t`) are shown. All these data structures are identified by a name (`fname` for the two files and simply `name` for the `sankey_t` type) and an initialisation status flag (`initialised`). Type `sankey_t` (the simplest among them) encapsulates the necessary `csv_t` and `R_script_t` types alongside methods to set the Sankey diagram from an instance of type `plant_t` (`set`), to create the final output (`exec`), and to check the initialisation status of the object (`check_init`). Types `R_script_t` and `csv_t` represent files and then include a unit number (`funit`) and logical flags necessary to check if the file is initialised (`initialised`) or open (`fopen`). Among the relevant type bound procedures (mostly with simple functionalities and self-explanatory names) there are two notable ones. The `R_script_t` `write_sankey` routine that generates the R script based on a CSV file (already created) and a specified category (i.e. material or energy) of the flows. More complex is the implementation of procedure `write_ecast_sankey` that is bound to type `csv_t`. This routine traverses the entire data structure of type `plant_t` to generate a CSV file with the following structure: source, target, value, category, unit of measure.

To accomplish elegantly the goal to traverse the entire data structure contained in an instance of `plant_t`, the `traverse` type bound procedure is invoked. This

routine has the advantage to implement only once the algorithm necessary to explore the data structure, independently of the specific task required that will be executed using a polymorphic mechanism.

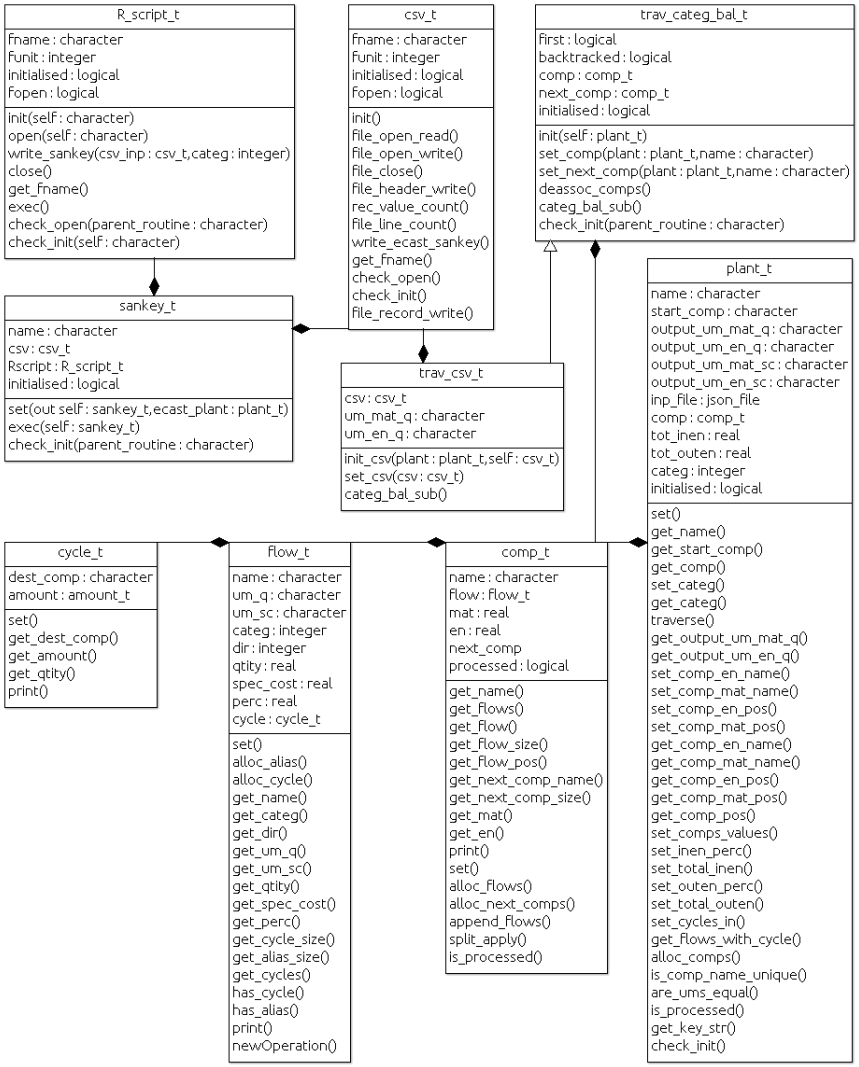

Fig. 5. Minimal UML class diagram describing the Sankey module implementation (obtained by ForUML [26]).

In this specific case, while traversing the data structure, it is necessary to operate on an instance of type `csv_t` that is not present in the scoping unit of `plant_t` and its `traverse` procedure. However, the `traverse` method uses also the `trav_cat_bal_t` data type (implemented in the same scoping unit) that contains the minimal information to explore the entire `plant_t` data structure. Extending type `trav_cat_bal_t` with the specific information necessary to generate the CSV file, it is possible to generate the super-class `trav_csv_t` that can be invoked with a polymorphism by the `traverse` routine algorithm. Figure 5 contains information also about this rather complex (but very effective and elegant) mechanism available in modern Fortran.

5 Industrial implementation scenarios

A number of options to integrate the computer program with traditional, existing manufacturing system tools are available. In this context, additional advantages that go beyond the visual representation of data become accessible. Six main areas are identified among these options (Table 1).

An interface to specialised Computational Fluid Dynamics (CFD) codes used to model the casting phase can provide an additional level of detail to the "overall picture" represented by the tool. Hence, it becomes possible to explore different design options or improve an existing product if enough data is exchanged in a standardised fashion. Work is currently undergoing to achieve these capabilities. One step further in this direction would be the implementation of automatic optimisation functions. To assist in the improvement of an existing product, the tool requires access to audited data. Alternatively, the design of a new product would require to access a database of manufacturing processes.

Table 1. Main features of the scenarios for the integration of the tool with existing manufacturing systems.

Scenario	Input	Additional benefit
Production improvement	Audited data	Accurate specifications (via interfaced tools, e.g. CFD)
Product design	Manufacturing processes database	Accurate specifications (via interfaced tools, e.g. CFD)
Benchmarking	Reference plants database	Basic Pareto analysis (i.e. find "low-hanging fruits")
Process monitoring	Real-time data	Process monitoring tool (via Internet of Things)
Training	Real-time data	Personnel didactic tool (via Internet of Things)
Life cycle assessment	Materials life cycle database	Product life cycle analysis

In a similar way, using a database of reference foundries, it is possible to benchmark the existing performance identifying the improvements with the highest return on investment (a basic Pareto analysis).

Feeding the software with real-time data from the production equipment, it becomes possible to monitor the process and also to control it, if a two-way communication channel is established. This last case can be easily imagined in the context of a "smart foundry" where the concept of "Internet of Things" is applied. Networking and suitable protocol communication capabilities would need to be implemented to achieve this goal. Furthermore, the availability of real-time data, transforms the computer program also into an effective training tool able to educate the workforce towards a virtuous behaviour including, in this way, also the "human" factor and completing its spectrum of action to the entire factory.

Finally, thanks to the modular design of the computer program, it is possible to implement embodied energy or CO_2-footprint flows that would allow the use of data provided by a materials life cycle database. In this way, a complete life cycle analysis of the product becomes possible.

6 Validation

The capabilities of the computer program (including the automatic generation of Sankey diagrams) are validated considering a complete, generic casting process that reproduces most of the key features that can be found in a foundry and it is based on industrial data producing aluminium alloy products.

The PFD (Figure 6) shows the process steps with light blue rectangular boxes, the material flows in yellow septagons and the energy flows in red elliptical shapes. Every node of the diagram reports its content and, in particular, the process step boxes show the calculated material content at the *end* of the phase. If the calculated material or energy content of any process step is negative (the obvious result of a wrong input data set), execution halts and an error message is issued. The user-defined unit of measure of the flows are kg for materials and GJ for energy. As mentioned in Section 4.1, an error is issued if an inconsistency in the units of measure is identified. Other errors related to missing data in the input file are identified at parsing-time and a relevant error is returned. The program calculates and shows also the contribution in percentage points of each input and output energy flow compared to the total (respectively, total input and output) to highlight quickly the main flows.

In this example, values are normalised for 1000 kg of melt material. Metal is initially heated until it changes phase in a furnace that is the main contributor to the energy input in the entire process and also is the place where material removed at later stages is collected to be re-melt. This step is also a major contributor of the energy losses (accounting for about 34% of the total energy output).

The liquid metal is then transferred to another furnace where it is put on hold to accommodate different production rates and let the metal oxides float to the surface. During this step another significant amount of energy is spent (more than a third of

the total) to maintain the temperature constant, with significant losses in sensible heat (about 45% of the total).

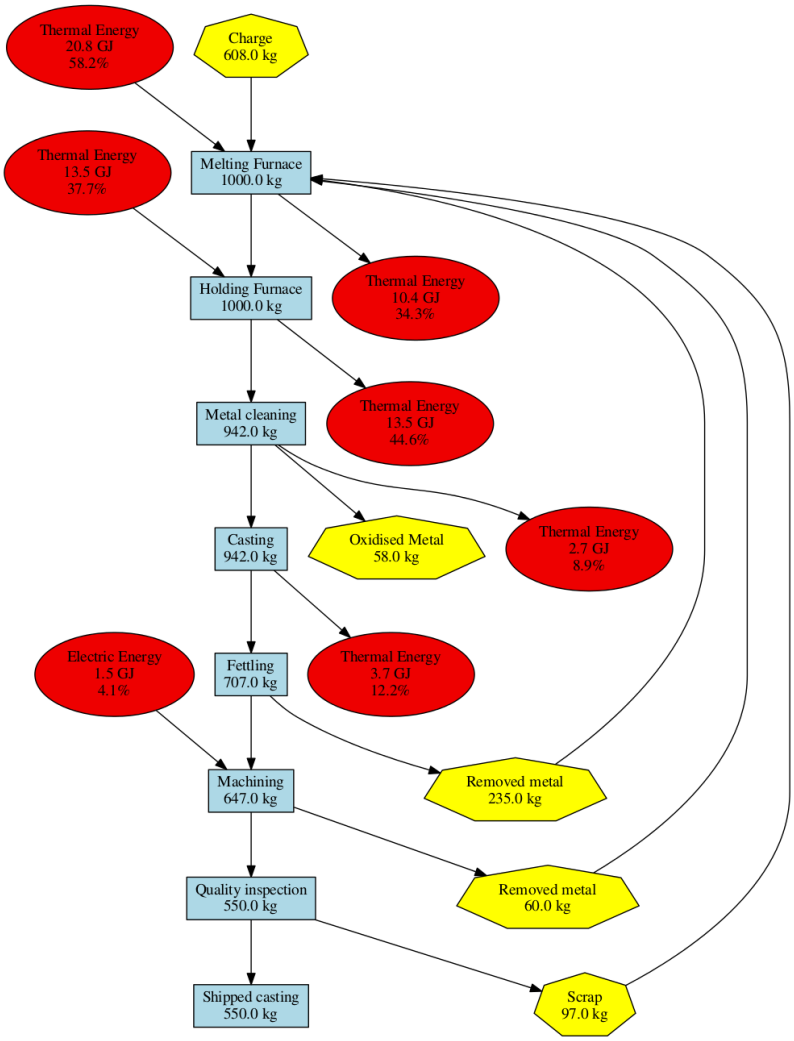

Fig. 6. Process Flow Diagram of the casting process used to validate the tool.

Before pouring, metal is cleaned (in this case, to remove oxides that inevitably form at high temperature) to improve the quality of the castings. Hence, the metal cleaning phase is associated with both material and energy (again, in the form of sensible heat) irrecoverable losses of medium-small proportions.

During the casting phase metal is poured into the mould cavity and cools down, generating another outgoing flow of energy (about 12% of the total outgoings).

After solidification, the product is then fettled with the removal of the gating and runners that are internally re-cycled in the melting furnace and account for a significant amount of metal (235 kg).

To achieve the required surface finish and dimensional accuracy, machining operations are required and generate additional metal scrap (60 kg) that is re-melt in the furnace. The energy input required during the machining operations is small compared to the other phases (about 4% of the total).

A quality inspection (that involves negligible flows of energy) is finally performed before shipping and the defective products (97 kg) are scrapped and returned to the melting furnace.

The relevant material and energy Sankey diagrams are presented in Figure 7. The figure is a screen-shot because the standard output is an interactive chart accessible by any web-browser where nodes can be dragged within the canvas and a mouse-over action highlights the flows and reports its numerical value in the relevant tooltip.

Finally, it should be noted that the program offers wide flexibility about the degree of accuracy to describe the foundry process and it is dependent on the information included in the input file describing the foundry. It is possible to span from a broad, simplified, lumped approach to keep the diagram and description of the plant simple, up to a fine-grained visualisation based on very detailed information (an example of a more accurate description of a foundry plant can be found in [20]). The option to build-up progressively a more and more accurate model when additional information become available, is another advantage of the design of this tool.

7 Conclusion

A program designed to understand more accurately foundry processes and to promote the development of more energy efficient casting practices has been presented. In particular, the recently introduced ability to automatically generate Sankey diagrams has been described and validated. This effective representation helps identify main areas of improvement, discover synergies or opportunities to recover waste energy, benchmark processes or improve personnel behaviour.

Several promising options are available for the integration of this tool with legacy manufacturing systems, bringing additional benefits besides data visualisation to reach the mentioned aims. Such options have been briefly outlined and, in particular, the integration of CFD codes is being actively explored.

Efforts to make the tool more user friendly to set-up are focussed to creating a Graphical User Interface (GUI) that can generate a suitable input file.

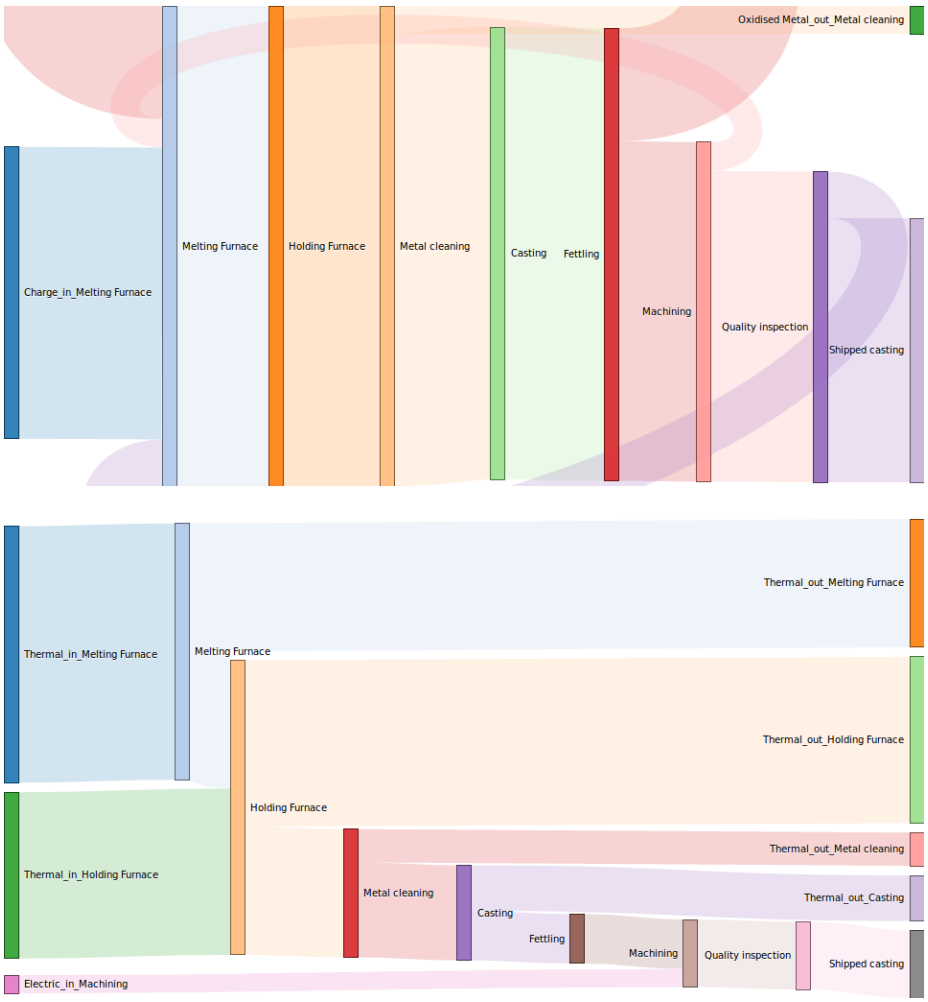

Fig. 7. Material (top) and energy (bottom) Sankey diagrams of the casting process used to validate the tool.

References

1. Salonitis, K., Stavropoulos, P.: On the integration of the CAx systems towards sustainable production. Procedia CIRP 2013;9:115–120.
2. UN General Assembly. Report of the World Commission on Environment and Development: Our Common Future. United Nations; 1987.
3. Elkington, J., Cannibals with Forks: the Triple Bottom Line of 21st Century Business, Capstone, 1997.

4. Mehrabi, H., Jolly, M., Salonitis, K.. Road-Mapping Towards a Sustainable Lower Energy Foundry. Foundry Trade Journal International 2016;190(3732):52–53.
5. Jolly, M.. Reducing energy in manufacturing – and how it affects the foundry industry. Foundry Trade Journal International 2016;190(3735):144–145.
6. Zeng, B., Jolly, M., Salonitis, K.. Manufacturing cost modelling of castings produced with CRIMSON process. In: TMS Annual Meeting. 2014, p. 201–208.
7. Salonitis, K., Zeng, B., Mehrabi, H.A., Jolly, M.. The challenges for energy efficient casting processes. Procedia CIRP 2016;40:24–29.
8. Klugman, S., Karlsson, M., Moshfegh, B.. A Scandinavian chemical wood pulp mill. Part 1. Energy audit aiming at efficiency measures. Applied Energy 2007;84(3):326–339.
9. Salonitis, K.. Energy efficiency assessment of grinding strategy. International Journal of Energy Sector Management 2015;9(1):20–37.
10. Cagno, E., Trucco, P., Trianni, A., Sala, G.. Quick-E-scan: A methodology for the energy scan of SMEs. Energy 2010;35(5):1916–1926.
11. Anderson, S.T., Newell, R.G.. Information programs for technology adoption: the case of energy-efficiency audits. Resource and Energy Economics 2004;26(1):27–50.
12. Thollander, P., Ottosson, M.. Energy management practices in Swedish energy-intensive industries. Journal of Cleaner Production 2010;18(12):1125–1133.
13. Schmidt, M.. The Sankey Diagram in Energy and Material Flow Management. Part II: Methodology and Current Applications. Journal of Industrial Ecology 2008;12(2):173-185.
14. Viere, T., Stock, M., Genest, A.. How to Achieve Energy and Resource Efficiency: Material and Energy Flow Modeling and Costing for a Small and Medium-Sized Company, Energy-related and economic balancing and evaluation of technical systems – insights of the Cluster of Excellence eniPROD, 2013, Proc workshop of cross-sectional group 1 "Energy related technologic and economic evaluation" of the Cluster of Excellence eniPROD, Wissenschaftliche Scripten, Auerbach.
15. Subramanyam, V., Paramshivan, D., Kumar, A., Mondal, Md. A. H.. Using Sankey diagrams to map energy flow from primary fuel to end use. Energy Conversion and Management 2015;91:342-352.
16. Herrmann, C., Zein, A., Wits, W.W., van Houten, F.J.A.M.. Visualization of environmental impacts for manufacturing processes using virtual reality. 44th CIRP Conference on Manufacturing Systems, Volume: Madison, USA: University of Wisconsin.
17. Gansner, E.R., North, S.C.. An open graph visualization system and its applications to software engineering. Software - Practice and Experience 2000;30(11):1203–1233. URL: www.graphviz.org.
18. Bostock, M., Ogievetsky, V., Heer, J.. D3: Data-driven documents. IEEE Trans Visualization & Comp Graphics (Proc InfoVis) 2011.
19. D3: Data-Driven Documents; accessed 2nd of June 2016. URL: https://github.com/d3/d3.
20. Pagone, E., Jolly, M., Salonitis, K.. The Development of a Tool to Promote Sustainability in Casting Processes. Procedia CIRP 2016;55:53–58.
21. R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing; Vienna, Austria; 2016. URL: https://www.R-project.org/.
22. rCharts – Interactive Charts from R; accessed 14th of November 2016. URL: http://ramnathv.github.io/rCharts/.
23. igraph – The network analysis package; accessed 14th of November 2016. URL: http://igraph.org/.
24. D3 Plugins; accessed 14th of November 2016. URL: https://github.com/d3/d3-sankey.
25. D3-plugin-captain-sankey – Friendly (subtree) fork of the "sankey" plugin; accessed 14th of November 2016. URL: https://github.com/soxofaan/d3-plugin-captain-sankey.

26. ForUML – Extracting UML Class Diagrams from Object-Oriented Fortran; accessed 14th of November 2016. URL: https://github.com/t2time/ForUML.
27. Chapman, S.J.. Fortran 95/2003 for Scientists and Engineers. McGraw-Hill Education; 2007.

Acknowledgements

The authors would like to acknowledge the UK EPSRC project "Small is Beautiful" for funding this work under grant EP/M013863/1.