Optimization of Mechatronic Systems for supporting Early Design Decisions

Daniel Malmquist, Daniel Frede, and Jan Wikander
Mechatronics Lab, Department of Machine Design
KTH Royal Institute of Technology
Stockholm, Sweden
malmqui@kth.se, frede@kth.se, janwi@kth.se

***Abstract*—It is a daunting task to design a modern mechatronic system. The many engineering fields coming together as one create integration issues and raise difficulties in finding the optimum solution to a design problem. This paper contributes to solving parts of this problem by presenting a novel approach for supporting design decisions in an early phase of the product development process of mechatronic products. In order to support decision making for a broad range of mechatronic products, a framework is presented which is capable of handling arbitrary system configurations consisting of standard mechatronic components, e.g. DC motors and gearboxes. Given according component models, the framework can optimize suggested concept ideas with regard to for instance system volume or inertia. In order to quickly optimize and evaluate different concepts, it is advisable to keep the underlying component models simple, yet powerful enough to be used for typical mechatronic systems. The presented approach therefore makes analytic usage of component models without requiring time-consuming numerical simulations. The results of the paper indicate that the presented method has good potential, especially for evaluating concepts in an early design phase, and further research effort should be put into developing it.**

I. Introduction

In order to strengthen its market position, a company is required to develop innovative products over time. Therefore, it is crucial to investigate the market thoroughly and to try to come up with product ideas fulfilling these market needs. These ideas need to be worked on and refined into several design concepts. The concepts are to be evaluated and the most promising ones to be designed in some more detail.

This product development process is divided into multiple phases – the number of phases depending on the process model used – usually incorporating some type of planning, concept generation, concept selection, detail design, and industrialization. Among the popular models of product development processes is the one given by Ulrich and Eppinger [1] which splits up the process into six phases. Other models are the four-phase process described by Pahl and Beitz [2] as well as the therefrom derived model presented in the guideline VDI 2221 [3]. As one possible representation, Fig. 1 shows the model as given in [1].

Ulrich and Eppinger [1] define a concept as a description of the form, function, and features of a product which is usually accompanied by a set of specifications, an analysis of competitive products, and an economic justification of the project. According to Pahl and Beitz [2] a concept or principle solution is the result of a conceptual design phase where the essential problems are abstracted, function structures are established, suitable working principle are sought for and these principles are then combined into working structures.

In [4], different methods of concept generation are explained and their advantages/disadvantages are discussed. It is common understanding [1], [2] that choosing the most promising concepts is crucial for the further product development, thus crucial for the success of the product and so for the company itself as 60-80% of the cost is spent during the concept phase [5]. In order to pick the best concepts, designers and engineers can benefit from a large pool of well-developed methods and tools. The work presented in this paper describes a new model-based method for evaluating mechatronic system concepts in order to pick the most promising ones for further investigation.

One of the difficulties in designing mechatronic systems is the fact that such systems incorporate various classic but also more recent engineering disciplines. Particularly therefore, but not limited to mechatronic systems design, the team of designers usually consists of members with different

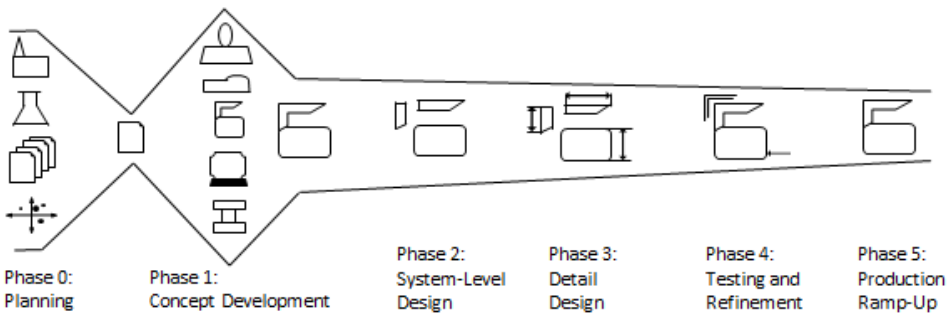

Figure 1. Generic product development process model adapted from [1].

background such as engineering experts from certain fields as well as non-engineering experts, such as marketing personnel which bring in feedback from costumers. A case study conducted by the authors revealed that decisions early in the design process could gain considerably from the availability and use of better supporting software.

In this case study, engineering companies in different fields of application were interviewed about the flow and process of their concept development phase. See TABLE I for characteristics of the companies. The size of the surveyed companies ranges from small business with a workforce as small as 15 employees up to global enterprises with as much as 35,000 employees. Given the diversity in both company size and type of products, the conclusions drawn from the study can speak for a broad range of engineering companies. The main focus of the interviews was on how the companies evaluate early concepts and what kind of tools, if any, are used for modeling and analysis of concepts.

TABLE I. COMPANIES INCLUDED IN THE CASE STUDY

Company	Employees	Notes
A	15+	University spin-off in aviation industry
B	100+	Subsidiary of global provider of motion systems
C	500+	Developer of high-tech manufacturing equipment
D	2000+	Automotive industry supplier
E	35000+	Automotive industry manufacturer

A couple of conclusions about simulation and modeling based concept evaluation can be drawn from the results of the study. All companies stated that the time it takes to model, analyze (e.g. simulate) and evaluate a concept is of high importance. This might seem obvious out of a cost perspective, but it is also important due to other reasons, such as the interesting statement that even five minutes of computation might be too long for a model evaluation since the user does something else during that time, hence losing his or her focus of the task at hand.

Case study results also showed that even large companies consider it too expensive to run multiple concepts in parallel, hence the ability to quickly evaluate concepts with computer aided modeling and analysis becomes vital. However, for some fields of engineering, modeling might not be possible due to reasons such as limited understanding of the physics involved.

Based on the case study results it can be seen that there is a lack of simple, fast, yet powerful, and well integrated tools available. A tool which could fill this gap would support engineers to make the right concept choices. Such a tool would most likely not be a generic solution which fits every design problem. The demand for quick set-up and evaluation of concepts will most likely result in constraints on the tool which when realized results in compromises in terms of model accuracy. These simplifications would in most cases be acceptable as long as they are reasonably made. Although every member of the concept development team is an expert in his/her very field, the common denominator in knowledge among them might not be sufficient to properly evaluate a concept. As an example, considering a rotating shaft, anyone can describe its radius, length and mass in meters and kilograms respectively, but few, except an expert in the field, have a proper sense of the combination of these physical quantities, kilogram times metre squared, which describes the shaft's inertia. Therefore the tool should rather need easily understandable properties as input, such as the shape and material of the rotating shaft instead of its inertia.

This leads to the consequential idea of a tool supported modeling and analysis method which can support in the concept phase by aiding in setting up models and quickly evaluating them. As mentioned, the team generating and working with concepts consists of experts with different backgrounds. Therefore, it is also important to make the potential method and tool easy to use, without the need for any deeper domain specific knowledge. A key benefit of such a tool would be the ease of communicating design ideas within a heterogeneous design team. As such, the tool can be seen as interpreter between the involved people with their diverse languages to describe concepts. Fig. 2 illustrates the setting of the mentioned supporting software tool.

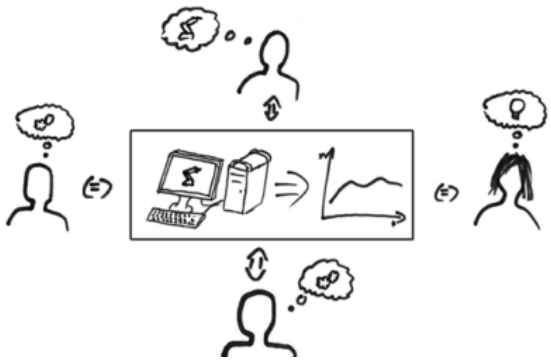

Figure 2. Idea of a simplistic tool supporting design decisions.

This paper presents a new method as well as software framework supporting it, which is intended to fill some of the mentioned gaps, e.g. ease of use and quickness of evaluation.

II. RELATED WORK / PREVIOUS WORK

A mechatronic design process is complex and extensive. As stated above, focus of the presented work is put on supporting the evaluation of early concepts by means of a tool-supported and model-based but easy to use method which also contributes to communicating design ideas and concept properties among the people involved. The preceding important task of generating reasonable concepts requires experience and broad knowledge in different domains. Also, as is common understanding, a structured method facilitates the process of generating promising concepts considerably. Literature, e.g. [4], lists a vast number of different methods for succeeding in developing such concepts. Similarly, a structured approach is also recommended for selecting the most promising concept ideas for further analysis and design. Again, standard literature on design theory, e.g. [1] and [2], covers well-tried methods for deciding among different solution concepts. By surveying the most relevant research

work in mechatronic design, in [6] different challenges related to design of mechatronic products are identified. The most important ones are pointed out as the lack of a common language to represent a concept, the difficulty in assessing consequences of selecting between two alternatives, and the transfer of models and information between domains and/or expert groups. The work described in [7] is interesting in regard to the fact that it integrates SysML with a process integration and design optimization framework (namely Phoenix Integration's ModelCenter). In doing so, the designers are assisted in more easily conducting various analyses which are required as the complexity of the systems to be designed has grown significantly. However, with its powerful interaction between multiple design tools, this approach rather aims at detailed design instead.

In [8], Roos developed a methodology for designing mechatronic servo systems in an integrated manner; that is treating the system as a whole during design and optimization. For the work presented in this paper, the underlying model-based concept needed for early system modeling and subsequent optimization is based on the earlier work by Roos et al. [8], [9]. Major contributions of his work are the component models for a motor driver, the electric machine itself (in particular permanent magnet machines), and an attached gearbox (both spur gears and planetary gears). It is assumed that the load driven by the servo system is known and well defined. Roos' methodology uses that load information in conjunction with a required load profile describing load motion and its corresponding torque to derive an optimal servo system for the specified load; optimal in regard to various criteria which are of interest for conceptual design and evaluation. Roos' main focus is on weight and size optimization, but based on that he is to some extent also able to design an optimal mechatronic servo system with respect to performance, efficiency, or cost. The system which Roos' method can handle is limited to a fixed set-up, which is illustrated in Fig. 3. However, the different component models created within his work are reapplied for the presented approach.

Besides the ability to design and optimize the physical system, Roos also discusses designing and optimizing the control system for the mechatronic servo system for which case the component models are extended to also capture system dynamics. Regarding this latter aspect, his method assumes rather detailed modeling of dynamics and involves numerical simulations. Of course, feedback and feed forward control solutions are at the heart of robust solutions to mechatronic design problems, and they determine to a large extent the merits of a particular product concept. This means that control performance aspects must be part also of early concept decisions. Currently, there are many control design methods available for robust solutions to mechatronic control problems, and many of these are supported by powerful computer aided tools. To meet the requirements on ease of use and short analysis time during early design stages, the use of such methods and tools as well as the corresponding numerical behavioral simulation are though problematic. However, many control related performance aspects can be predicted and analyzed directly based on the system model of the physical concept (to be controlled), without requiring numerical simulations. A linear system model in the form of a multiple-mass oscillator chain serves this purpose for many mechatronic systems [10]. Examples of control related aspects which are conveniently and analytically derived from such a model (given selected actuation and sensor configurations) are resonance and anti-resonance frequencies, poles, zeroes and aspects related to collocation/non-collocation of sensor and actuator.

There is a large number of available modeling, design, and simulation tools aimed both for rather general purposes (e.g. Modelica) and for use within very specific application fields (e.g. MCAD). It is therefore unreasonable to here elaborate on all the numerous approaches and their advantages and disadvantages. Instead, it is highlighted that most tools are intended either for rather detailed design or for rather high level system design (e.g. SysML). In the first case, the modeling is done at a high level of detail at the expense of analysis time and set-up complexity. In the latter case the system representation typically becomes too abstract to support common understanding between domain engineers and also to support model based analysis of key properties of the design concepts to be evaluated.

Looking into the various simulation tools in more detail, one sees that there are different approaches of how to model and simulate a given system. The more traditional way is to define dependent and independent variables as part of the model, and then have the specific model equations set up by ordering those variables accordingly. A newer approach represents a model through algebraic and differential equations. The model can be used in a flexible manner, by having the underlying equations solved for the dependent variables specifically needed for a certain simulation task. Although this approach is often considered to have the most beneficial properties, one must also take into account the expertise needed to set-up these types of simulations. Further, the advantage of being able to re-use one and the same model equations for various tasks is achieved at the disadvantage of being limited to equation based models only. In contrast, fixed causal models are less flexible but offer the possibility to also have non-equation based models.

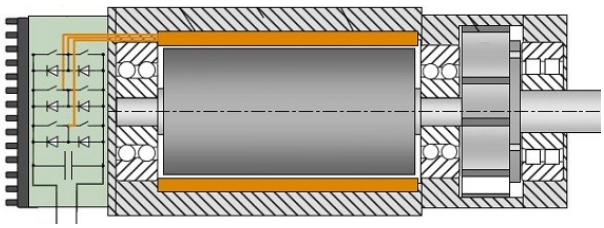

Figure 3. Example of servo system which can be optimized by Roos' methodology [8].

III. RESEARCH GOAL

On a longer term, the goal of this research is to develop a new model based design methodology and supporting tool for mechatronic product development. The method should be easy to use and able to support design decisions in an early product development phase. To meet the requirements on ease of use

and short analysis time, the use of behavioral simulation should be avoided. For fair comparison of different design concepts the method should include system level optimization. Since early design decisions as well system level optimization are in focus the method should only require a limited level of modeling detail. The new method should work with arbitrary system configurations and be flexible in terms of possible optimization goals. Prime properties in focus for optimization and analysis are concept size, weight, energy/power consumption and control performance. The supporting tool solver should be reasonably fast so that the operator does not lose focus of the design task at hand.

The paper describes initial work on the new methodology and this work is validated by optimizing the same system as Roos describes in his research [8], i.e. DC motor, planetary gearbox and load.

IV. Method and Models

A. General

The method described in this paper differs from many conventional methods for concept evaluation in the sense that it does not rely on actual simulation. Instead, the component models describe properties as a function of key design variables and are evaluated analytically. This is due to the fact that involving simulation and analysis of simulation results in the design optimization loop would drastically increase the evaluation time of a particular system. Examples of captured relations could be component dimensions to weight and load bearing capacity, or material choice to component stiffness.

In difference to methods like the one described in [7], the method can be applied directly after informal sketching of a system without the need to do detailed modeling of components. This allows for time efficient concept evaluation and makes it perfect to use while evaluating concepts in an early design phase. Another difference, which allows easier use, is that no interfacing between tools is required to use the method. However, the modular definition of component models in the framework supports use of external component models if a higher level of detail is required for a particular case.

A key aspect of the method is the design of component models to facilitate analysis of the requested properties and to do this at a level of detail which fits early concept evaluation. To allow an arbitrary component order in the system the component models are defined as independent modules. The load used as dimensioning constraint for each component is calculated in a manner similar to flow and effort based modeling. However, instead of dynamically calculating the load over time, static values such as the peak and root mean square (RMS) are used. For system optimization, genetic algorithms are used, but also other optimization methods could of course be applied.

B. Models

In earlier work [8], [9], component models where developed which describe the physical properties with static equations to facilitate early system optimization. For these models, the level of detail is kept limited with the advantage of much easier handling of the equations.

A few of these models are used here to exemplify the use of the new method and also for the purpose of verifying the results against the work of Roos. The system concept used as example is an actuation system, including a DC motor and planetary gearbox, which is to be optimized for volume. Therefore, the following models are derived for the volume only.

The DC motor model used in this work is derived in [8]. The volume of the DC motor can be approximated by a cylinder, given as

$$V_m = \pi r_m^2 l_m \tag{1}$$

Consequentially, the motor's rotor length l_m and the radius of the stator r_m need to be known. The dominating constraint for dimensioning the motor of a servo system is to avoid overheating of the winding insulation. Therefore, the machine's continuous torque rating $T_{m,rated}$ must be higher than the RMS of the machine torque $T_{m,RMS}$, as expressed by

$$T_{m,rated} \geq T_{m,RMS} \tag{2}$$

According to Roos, the rated machine torque is given by

$$T_{m,rated} = C_m l_m r_m^{2.5} \tag{3}$$

where C_m is a constant for a specific machine type and for the same cooling conditions. Equation (3) neglects the longitudinal heat transfer within the motor and also assumes a constant magnetic flux density.

Assuming a stiff system, the required motor torque T_m to drive a specific load is given by

$$T_m = \left(J_m + J_g + J_0\right)\ddot{\varphi}_m + \frac{T_l}{n \cdot \eta_g} \tag{4}$$

where $\ddot{\varphi}_m$ is the angular acceleration of the motor shaft, J_m is the mass moment of inertia of the motor, J_g is the gearbox inertia, and J_0 represents all other inertias in the system, such as bearings, resolver, and shafts. These inertias are given with respect to the motor side. The second summand is the load profile's torque T_l on the load side transferred onto the motor side via the gearbox with a given gear ratio n and total efficiency η_g.

With (4) it follows that the RMS value of motor torque $T_{m,RMS}$ can be stated as

$$T_{m,RMS} = \sqrt{\frac{1}{\tau}\int_0^\tau \left(\left(J_m + J_g + J_0\right)\ddot{\varphi}_m + \frac{T_l}{n \cdot \eta_g}\right)^2 dt} \tag{5}$$

where τ is the cycle time of the load profile. Furthermore, the inertia J_m of any motor can be expressed as a function of machine radius and length, given as

$$J_m = C_{mj} l_m r_m^4 \tag{6}$$

Both the constant C_{mj} and the earlier mentioned constant C_m are calculated based upon existing motors, that is they are derived by solving (3) and (6) for C_m and C_{mj}, respectively,

where all other variables are known. Eventually, (6) is put in (5), as well as (5) and (3) are put in (2). The resulting equation

$$C_m l_m r_m^{2.5} = \sqrt{\frac{1}{\tau}\int_0^\tau \left(\left(C_{mj} l_m r_m^4 + J_g + J_0\right)\ddot{\varphi}_m + \frac{T_l}{n \cdot \eta_g} \right)^2 dt} \qquad (7)$$

is then solved for the motor length l_m, resulting in a function of l_m depending on stator radius r_m. In conjunction with a form factor constraint for the motor, such as

$$0.5 \leq \frac{l_m}{r_m} \leq 5 \qquad (8)$$

the volume of the motor, (1), can be determined. Hence, a model for DC motors is developed which scales physical properties to new fictive machines based on properties of existing machines.

The second important component of a servo drive is the gearbox. Again, a detailed derivation of the model can be found in [8], whereas this work only focuses on three-wheel planetary gears since their volume, weight, and inertia are less compared to spur gear pairs for transmitting the same torque. In contrast to the motor model described above, the planetary gear is dimensioned for the maximum torque of the load profile as the number of load cycles over the gear's life span is assumed large and thus the total time of transmitting that peak torque must not be neglected.

Similarly to the motor volume, (1), the outer volume of the gearbox can be approximated as

$$V_g = \pi r_g^2 b \qquad (9)$$

where r_g is the outer gear radius and b is the total width of the gearbox. The design guidelines for spur gears SS 1863 [11] and SS 1871 [12] reveal that the gear size is limited by mechanical fatigue, the bending stress in the root of a gear tooth, and the Hertzian pressure at the teeth contact surfaces. However, if the number of sun gear teeth is small and/or the gears are made out of not very hard steel, it is likely to be solely the Hertzian pressure which limits the gear design. Without further explanation the following constraint must be fulfilled for every teeth surface in contact.

$$r_g^2 b \geq Z_H^2 Z_M^2 Z_\varepsilon^2 K_{H\alpha} K_{H\beta} \frac{T_l (n-1)^2}{6(n-2)\sigma_{H,max}^2} \qquad (10)$$

Applying the standard gear parameters defined in Roos' work, equation (10) simplifies and can be rewritten as

$$r_g^2 b \geq 4 \cdot 10^{10} C_{gr}^2 \frac{T_l (n-1)^2}{(n-2)\sigma_{H,max}^2} \qquad (11)$$

where $\sigma_{H,max}$ is the maximum allowed flank pressure which is derived by considering the maximum allowed Hertzian pressure for a particular gear material as well as a desired safety factor. The constant C_{gr} represents the relation between the outer gear radius and the gear's pitch radius. Equation (11) together with a form factor constraint which relates gear radius r_g to total gear width b is then sufficient to solve the volume function of the gear head (9).

A noteworthy improvement over Roos' implementation of the gear volume model is that the improved model takes into account a list of allowed gear modules. Hence the new gear model only calculates gearboxes which are actually feasible according to standardized gear modules. As a result, a quantization takes place as now the gear ratio does not only depend on teeth number and gear radius but also the module constraint must be fulfilled. In doing so, it is much easier to determine and optimize the distribution of the individual gear stages' ratios. The resulting difference in the gear volume of the example system can be seen in Fig. 4.

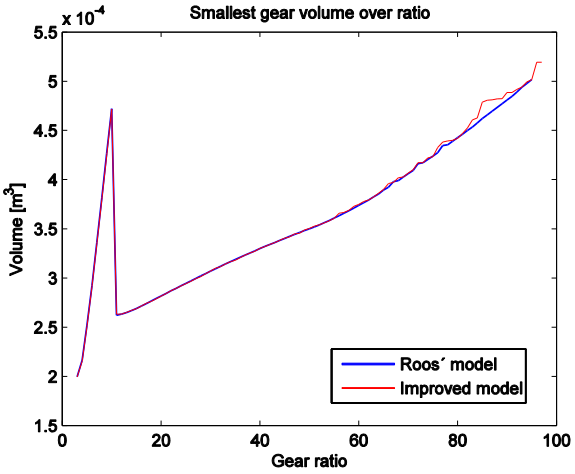

Figure 4. Gear volume of the example system.

V. IMPLEMENTATION

A. General

A prototype of the tool has been implemented as a MATLAB toolbox, with the main parts being a solver, the models and a language to describe system configurations. The implementation of these parts is described in this section.

An example system of a DC motor and a planetary gearbox is used and referred to as the *system*. The load profile used in the examples is shown in Fig. 5. The reason for using this system is the possibility to compare with results from the same example system as presented in [8].

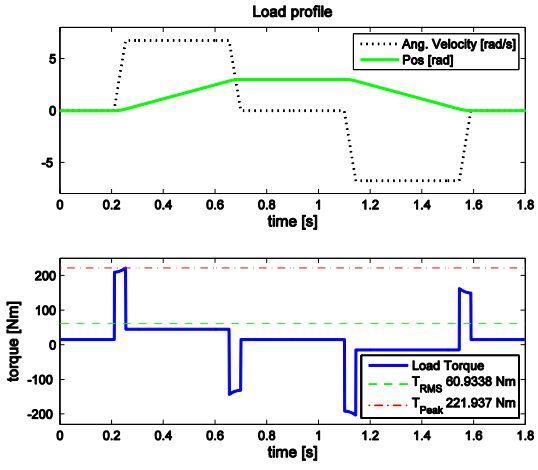

Figure 5. The nominal load profile used as example in this paper.

B. *System Description*

The system intended for optimization needs to be described in a language which can be interpreted by the design tool. The language needs to describe which components the system consists of, how they are connected, and what parameters they have. Languages such as SysML and Modelica could be used for this, they were however deemed unnecessarily complex for this prototype tool, with functionality far exceeding what was needed. A future tool could of course use one of these languages for compatibility with other tools. The XML-format is widely used and can with fair ease be implemented in most programming languages, thus allowing for simple porting of the tool to other programming languages and/or platforms.

For describing the system an XML-based format is applied where each system component is represented with a node directly under the XML root node. The component node's sub-nodes describe for instance component type, parameters, and which components are connected to its flanges. The component parameters, which can be either design variables or constant properties, can be defined either as a fixed value or as a value given by the optimizer. If the value is to be given by the optimizer, the parameter is defined either with an upper and lower limit or as a set of allowed values. A sample component node-tree can be seen in Fig. 6.

```
<component>
    <id/>
    <name/>
    <type/>
    <LoadFlanges> </LoadFlanges>
    <DriveFlanges> </DriveFlanges>
    <parameters> </parameters>
</component>
```

Figure 6. Example of an XML component node tree. Sub-nodes with both opening and closing xml-tags contain sub-nodes of their own.

C. *Solver architecture*

The solver works in a number of steps, as illustrated in Fig. 7. First of all the program is initialized by e.g. loading configuration files (components and setup of the system to be optimized among others), determining what parameters should be used for optimization, and reading in the optimization objective. The optimization objective is defined in the same XML file as the system configuration in the form of a string value in a node and can be constructed out of one or more of the inputs and outputs of the available component models in conjunction with standard arithmetics. An example optimization objective node can be seen in Fig. 8.

```
<objective>
    @@model@1@Volume@output@Volume@@ * -1
</objective>
```

Figure 8. Example of optimization objective node where the minimization objective is the *Volume* output of the *Volume* model for component *1* multiplied by *-1*.

After initialization is done, the optimizer is executed. The optimizer uses a special *Objective* function which generates an instance of the system with the parameters given from the *Optimizer,* and then evaluates the system for the specified objective. This is done by passing the optimization objective (like the one shown in Fig. 8) to an *Interpreter* function. The *Interpreter* function uses regular expressions to extract information about for instance what model should be called to get the right output and then runs the *Call Model* function to call it. The *Call Model* function figures out if all input data which has been defined in the component configuration file as required for calling the model is available. If not, it is due to that the model requires the output of another model as input and hence that model has to be run first. To accomplish this, the *Interpreter* function is called again with the input as the string to be interpreted instead of the optimization objective string. Hence the two functions, *Call Model* and *Interpreter*, call each other in a recursive manner until the sought output can be determined. See Fig. 9 for an example of a mechatronic servo where the model describing the volume of the motor is sought.

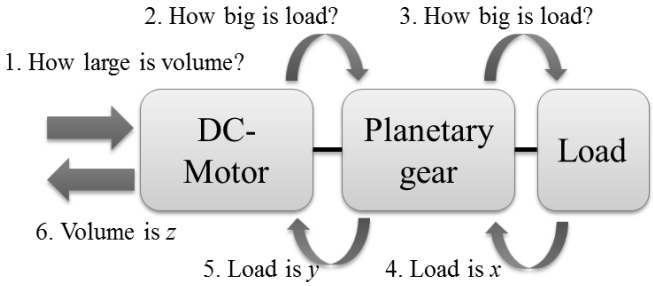

Figure 9. Example of a mechatronic servo where the *Volume* output of the model describing the *Volume* of the motor is sought. To calculate this value the load on the motor is needed which causes the load of the connected components to be calculated in a recursive manner.

To save computation time the result of a model call is stored and reused if needed again, therefore the same model need only be called once for each system instance, i.e. only once for each objective function evaluation.

Most non-gradient based optimization methods, such as Complex-RF [13] or evolutionary algorithms, would work,

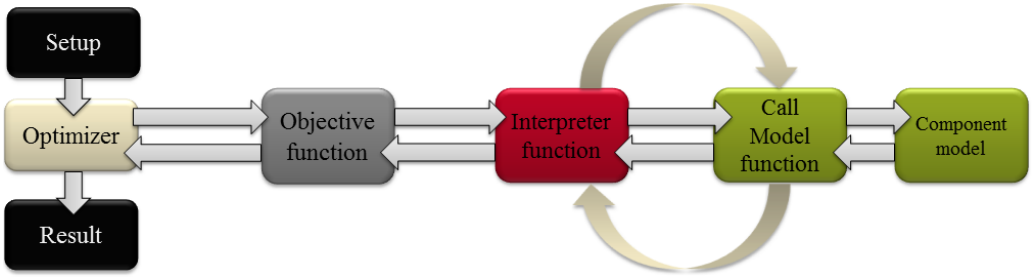

Figure 7. Prototype tool simplified program flow.

however for testing the prototype tool a genetic algorithm based optimizer was chosen and a single objective was used.

When the *Optimizer* function is done, the resulting system design variables are outputted to the user as well as used to generate a visual representation of the system in form of a sketch. An example output sketch can be seen in Fig. 10.

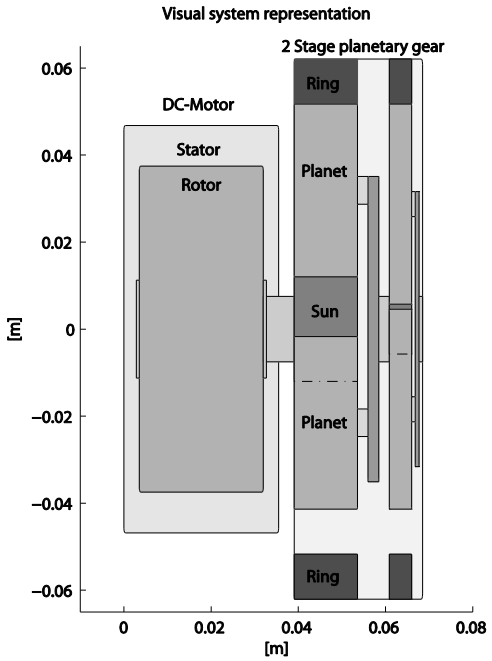

Figure 10. Example output sketch of a system consisting of a DC motor and a two-stage planetary gearbox.

VI. RESULTS

To validate the new model implementations and the ability to use them properly, the tool was used to calculate the volume of the example system for a range of design variable values instead of specific design variable values generated by the optimizer. The design variables used were gear radius and total gear ratio. As can be seen by Fig. 11 and Fig. 12 the results of the two implementations and methods are virtually the same.

A set of optimizations was run with different population sizes for the genetic algorithm to test performance and accuracy. The system optimized was the example system mentioned earlier and three design variables were optimized: total gear ratio, number of gear stages and gear radius. The optimized volume was compared with the optimized volume from Roos' method. TABLE II presents average, best, and worst results compared to those of Roos' method as well as the average time required for optimization over ten optimization runs for each of four different population sizes. The test runs were done on an average consumer-PC.

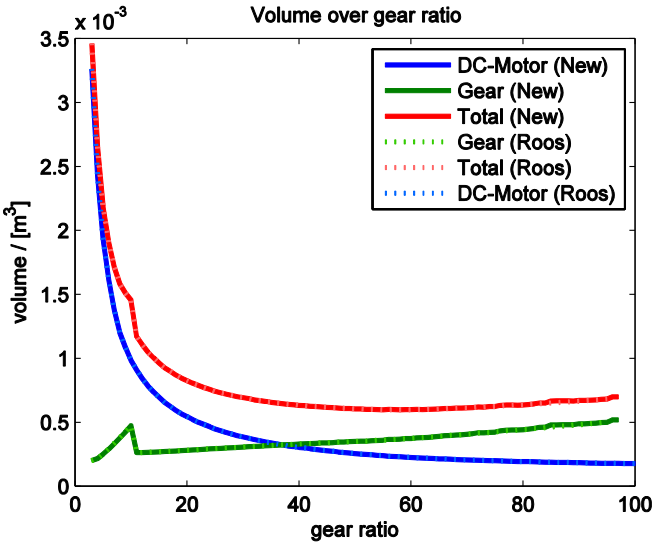

Figure 11. Total system volume for a range of gear ratios and the corresponding best gear radius.

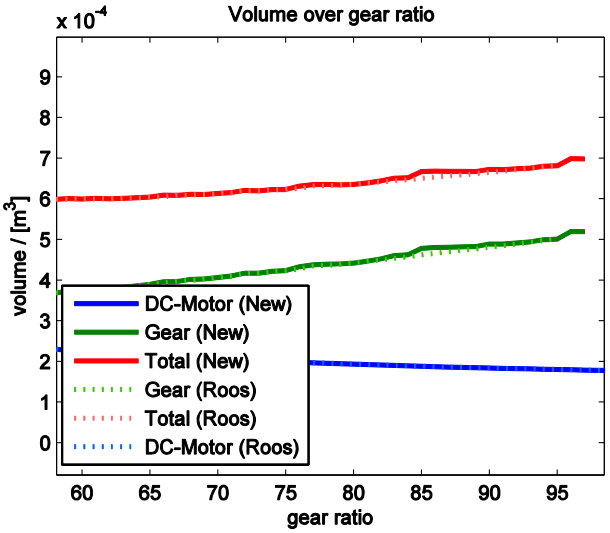

Figure 12. Zoomed in version of Fig. 11.

TABLE II. AVERAGE, BEST, AND WORST RESULTS FROM POPULATIONS BETWEEN 10, 20, 30 AND 40 INDIVIDUALS OVER TEN OPTIMIZATION RUNS EACH. HIT ACCURACY IS DEFINED AS THE RATIO BETWEEN NEW METHOD'S RESULTS AND THE RESULTS OF ROOS METHOD.

Population Size	Optimization time [s]	Hit accuracy [%]		
		Worst	Average	Best
10	19.2	1.8414	0.7272	0.3873
20	34.3	0.4290	0.3240	0.2037
30	56.9	0.4307	0.2639	0.2037
40	73.2	0.2938	0.2270	0.2037

VII. CONCLUSIONS

Tests have shown that the new method can replicate results close to identical to those of Roos' method. The new method should work with an arbitrary system configuration; this however is yet to be tested.

The optimization results presented in this paper show good results for all different population sizes. For population sizes 20, 30 and 40 the best achieved results are 0.2037 % worse than those of Roos' method. This could be due to that the new implementation of the planetary gearbox model only allows gear modules from a fixed list (usually a list of standard modules) whereas Roos' implementation allows arbitrary modules. The time required to optimize the system can also be considered as good enough, considering the five minute requirement mentioned earlier in this paper. The time required to do a single system evaluation is expected to scale close to linear with the amount of components in the system. However, the effects on the time it takes to optimize the system due to added design variables related to having more system components will likely not be linear. A more detailed analysis would be needed to draw proper conclusions about both optimization time and results considering that there are many more parameters than population size that affect the outcome. These results are only intended to give a rough estimate of achievable optimization run times and results. More types of optimization algorithms should also be tested.

This paper has focused on the underlying framework of a possible future optimization tool and neglected the need for the simple user interface aspect of the case study.

It has been shown that the method has strong potential to support design of new mechatronic systems. However, further development of the tool is needed.

VIII. FUTURE WORK

There are a lot of possibilities for future work on the new methodology and tool. The authors of the paper plan to for instance look further into developing models for new types of components, both mechanical and electrical. There is also a plan to extend the current components with models describing other properties than volume, such as weight and cost. Adding more properties for use in concept evaluation, it becomes natural to make the tool more powerful by adding algorithms for multi-objective optimization.

As mentioned in the related work section, many system properties related to dynamic behavior and control performance can be deduced analytically from a linearized dynamic model. Including such model properties in concept evaluation through multi-objective optimization would result in integrated control and process design taking place much earlier than what is currently the case. This will be an important part of future work.

Useful improvements could come from a deeper analysis of optimization algorithms and what optimizer configuration to use.

In general, further work is needed to demonstrate and test the capabilities of the new method and tool.

ACKNOWLEDGMENT

This research is funded by the Swedish Foundation for Strategic Research through the ProViking program.

REFERENCES

[1] K. T. Ulrich and S. D. Eppinger, "Product design and development", McGraw-Hill/Irwin New York, 2011.

[2] G. Pahl, W. Beitz, J. Feldhusen, and K. H. Grote, "Engineering design: A systematic approach", Springer-Verlag London, 2007.

[3] VDI – Association of German Engineers, "VDI 2221 - Systematic approach to the development and design of technical systems and products", Beuth Verlag Berlin, 1993.

[4] A. M. King., S. Sivaloganathan, "Development of a methodology for concept selection in flexible design strategies", Journal of Engineering Design, Vol.10, No.4, pp 329-349, 1999.

[5] A. H. B. Duffy, M. M. Andreasen, K. J. Maccallum, and L. N. Reijers, "Design co-ordination for concurrent engineering", Journal of Engineering Design, Vol.4, pp 251-261, 1993.

[6] J. M. Torry-Smith, A. Qamar, S. Achiche , J. Wikander, C. During, and N. H. Mortensen, "Mechatronic design – still a considerable challenge", in Proceedings of the ASME 2011 International Design Engineering Technical Conference & Computers and Information in Engineering Conference IDETC/CIE 2011, Wahsington, DC, USA, August 29-31, 2011.

[7] B. I. Min, A. A. Kerzhner, and C. J. J. Paredis, "Process integration and design optimization for model-based systems engineering with SysML", in Proceedings of the ASME 2011 International Design Engineering Technical Conference & Computers and Information in Engineering Conference IDETC/CIE 2011, Wahsington, DC, USA, August 29-31, 2011.

[8] F. Roos, "Towards a methodology for integrated design of mechatronic servo systems", Doctoral Thesis, Department of Machine Design, KTH Royal Institute of Technology, Stockholm, Sweden, 2007.

[9] F. Roos, H. Johansson, and J. Wikander, "Optimal selection of motor and gearhead in mechatronic applications", Mechatronics, Vol.16, No.1, pp 63-72, February 2006.

[10] K. Janschek, "Mechatronic systems design – methods, models, concepts", Springer-Verlag Berlin Heidelberg, 2012.

[11] Swedish Standard SS 1863, "Spur gears – geometrical data", edition 4, SIS - Swedish Standards Institute, 1978.

[12] Swedish Standard SS 1871, "Spur and helical gears – calculation of load capacity", edition 3, SIS - Swedish Standards Institute, 1978.

[13] P. Krus and J. Andersson, "Optimizing optimization for design optimization", in Proceedings of the ASME 2003 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2003, Chicago, USA, September 2–6, 2003.