Chapter 8
CAESES—The HOLISHIP Platform for Process Integration and Design Optimization

Stefan Harries and Claus Abt

Contents

S. Harries (✉) · C. Abt
FRIENDSHIP SYSTEMS AG, Potsdam, Germany
e-mail: harries@friendship-systems.com

A. Papanikolaou (ed.), *A Holistic Approach to Ship Design*,
https://doi.org/10.1007/978-3-030-02810-7_8

Abstract This chapter focuses on the approach taken within the European R&D project HOLISHIP to flexibly integrate and utilize software tools and systems of tools for the design, analysis, and optimization of maritime assets, primarily of ships. The tools and systems come from different developers, companies, and research institutes and, consequently, have been mostly used as stand-alone applications. The purpose of integration is to create (software) synthesis models that comprise many, if not all, key aspects that ought to be considered when working on a specific ship design task. Rather than proposing an all-encompassing single (monolithic) design system in a top-down approach, the idea pursued within HOLISHIP is to support bottom-up approaches, namely the ad hoc assembly of dedicated models that are fit for a specific purpose under the umbrella of a state-of-the-art computer-aided engineering (CAE) system, namely CAESES®. This CAE system will be elaborated in the present book chapter. The approach of tool integration will be discussed, and it will be shown how to replace time-consuming simulations by means of surrogate models. Examples taken from the design and optimization of a RoPAX ferry and of an offshore supply vessel will be given for illustration.

Keywords Process integration and design optimization (PIDO)
Computer-aided engineering (CAE) · Simulation-driven design (SDD)
Synthesis model · Surrogate model · Parametric model · Tool coupling

8.1 Introduction and Motivation

Design and optimization are closely connected. As soon as a team of engineers has realized a feasible design, i.e., something that works and fulfills all requirements, somebody starts thinking of how to come up with an improvement. Improvements are looked for because of competitive markets but also because people inherently strive to do things better (Nowacki in Birk and Harries 2003).

Maritime assets such as ships and offshore structures are very complex systems that operate in harsh environments. They are run by people, they shelter people, and they are meant to serve people. Consequently, many systems, sub-systems, and components need to be brought together in order to reach a design that is uncompromisingly safe, economically attractive, and environmentally friendly.

Most systems and sub-systems are very closely connected and typically need to be in a state of balance. The single most important (and most obvious) balance for a floating object is that its overall weight may not be larger than the weight of the water it displaces at its desired draft. Another balance needs to be established between the

propulsion system, including engines and power supply, and the energy it takes to operate the asset in all possible scenarios.

Due to the complexity of maritime assets, a great deal of experience is called for, and still many analyses are typically made one after another as idealized via the classical design spiral: A team of designers and engineers makes reasonable assumptions, undertakes analyses based on them, and subsequently corrects and refines the design. This is repeated until a single design or a small set of potential designs has been found, essentially constituting a sequential and iterative process. Since expert knowledge may not be internally available for all disciplines, external suppliers are regularly involved to support the design effort.

This traditional approach is particularly successful if an earlier project is at hand that is close to the new design task. It also leads toward substantial progress, at least incrementally and over time. For increasingly demanding and shifting markets, for completely new missions and for design challenges for which only little experience is (at least locally) available, however, this is often too slow and too cumbersome.

A different approach is therefore proposed in which many, if not all, important disciplines for designing a maritime asset are taken into account concurrently. This can only be achieved if many dedicated systems and tools for design, analysis, and simulation, as disparate as they may be, are closely combined to form an overarching computer-aided engineering (CAE) environment that

- holds, converts, and shares data
- controls interactions and logical dependencies, and
- supports the swift creation of variants both manually and automatically.

The European R&D project HOLISHIP sets out to establish such a CAE environment, the so-called HOLISHIP platform(s), by bringing together systems and tools as well as the expertise from different institutions and sites, typical of the many stakeholders in the maritime industry and their heterogeneous CAx solutions.

Instead of proposing an all-encompassing "super system," a rather moderate approach was taken, namely the flexible combination of legacy systems and tools as needed to solve a number of interesting application cases (AC) along with the possibility to add further tools quickly and efficiently, also beyond the original partners of the HOLISHIP consortium.

This chapter explains how process integration and design optimization are realized, what methods are provided, and how to benefit from the approach.[1]

[1]Naturally, it is hoped that this will add value to the creative and excellent design work that has been done over all the years since human beings have put to sea. In no way is the intention of this chapter to suggest that process integration and design optimization are the only ways to achieve further improvements.

8.2 Process Integration and Design Optimization

8.2.1 Overview

Quite a few commercial software systems for process integration and design optimization (PIDO) are available today. Typically, they are generic systems that provide several of the following techniques:

- Multi-tool integration
- Process automation
- Process capturing and reuse
- Design space exploration/design of experiments (DoE)
- Exploitation/deterministic and stochastic optimization
- Multi-objective and multi-disciplinary optimization (Pareto frontiers)
- Robust optimization and sensitivity analyses
- Visual data analytics
- Data mining
- Surrogate modeling
- Multi-fidelity and multi-physics modeling
- Simulation data management and
- Product life-cycle management

Walsh (2018) summarized the available systems under the general term of design space exploration. Several of the methods, for instance DoE and surrogate models, are considered by Bostrom (2014) to belong to the wider field of artificial intelligence (AI). One overarching theme of PIDO is that of enabling the efficient generation and systematic assessment of large sets of prototypes as discussed by Schrage (2000).

For the integration of systems and tools within HOLISHIP, CAESES® was chosen. This is because CAESES® offers, in addition to many of the PIDO techniques specified above, a comprehensive computer-aided design (CAD) package with which to model but also to convert geometry as needed to feed the various analysis and simulation systems. In this sense, CAESES® is a computer-aided engineering (CAE) environment that tightly combines PIDO and CAD.

8.2.2 Background

CAESES® has been developed and licensed by FRIENDSHIP SYSTEMS AG with its headquarter in Berlin/Potsdam, Germany. The company is a spin-off from the Technical University of Berlin, established in 2001, and provides systems and consultancy for simulation-driven design (SDD).

Ideas of CAESES®, particularly with respect to parametric modeling of hull forms, go back to R&D projects undertaken in the 1990s (Harries 1998), but the actual code has been developed from scratch, starting in 2004. A first release of CAESES® was

launched in 2007 (then named *FRIENDSHIP-Framework*). The description and the screenshots given here are based on CAESES 4.3 as released in 2018.

CAESES 4.3 comprises around 1,000,000 lines of written and more than 6,000,000 lines of generated code. It is mainly developed in C++, uses Qt for its cross-platform GUI (www.qt.io), combines a legacy CAD kernel by FRIENDSHIP SYSTEMS with several commercial CAD kernels, most notably the SMlib by Solid Modeling Solutions (www.smlib.com), links with DAKOTA by Sandia National Laboratories (dakota.sandia.gov) as an optimization kit and, furthermore, incorporates more than 20 open-source libraries. CAESES® can be run on both Windows® or Linux™. Furthermore, it allows the execution of software located remotely and in cross-platform environments.

8.2.3 *Overview of Intrinsic CAESES Functionality*

CAESES® takes a different approach than other PIDO environments. It not only provides various coupling mechanisms and a wide range of optimization strategies but also offers parametric CAD for robust models of variable geometry. The overall functionality is summarized in Fig. 8.1 (green boxes with blue components).

Originally, CAESES® was developed for the simulation-driven design of functional shapes that serve a fluid-dynamics purpose such as hull forms, propellers and energy-saving devices for ships as well as impellers, volutes, diffusers and manifolds for turbomachinery, and combustion engines. In a typical design and optimization

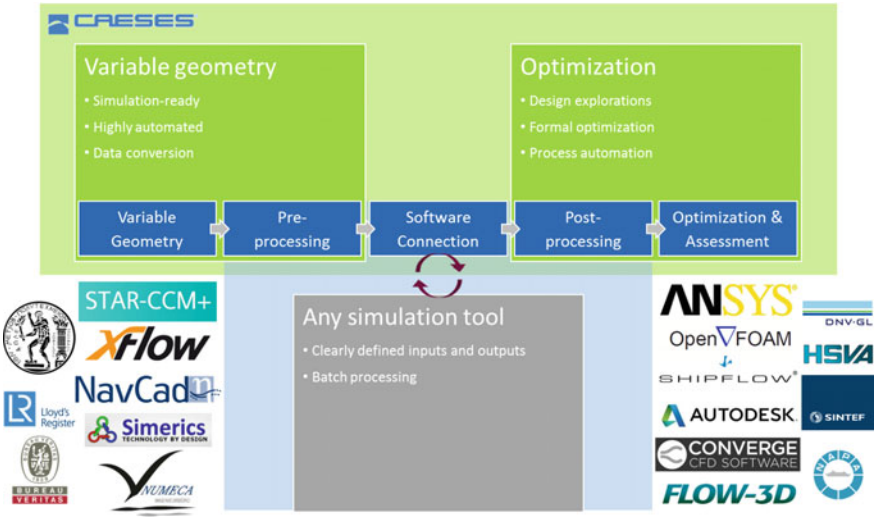


Fig. 8.1 Overview of CAESES® functionality along with a selection of software systems frequently coupled and providers of tools and systems from the HOLISHIP consortium

process, several components (the so-called big five of CAESES® as shown in Fig. 8.1) are brought together:

1. Variable geometry: A parametric model is developed, and a shape variant is created as an instance of the chosen parameter set.
2. Pre-processing: The variant is pre-processed to enable the simulation(s) to take place.
3. Simulation(s): For all variants of interest, one or several simulations are undertaken.
4. Post-processing: Variants and their data are post-processed (e.g., visualizing flow fields for comparison).
5. Optimization and assessment: Variants are produced and assessed in accordance to the selected optimization strategy (e.g., Sobol, MOGA), repeating the sequence from variable geometry to post-processing again and again.

With its various CAD kernels, CAESES® provides both boundary representation (BRep) and constructive solid geometry (CSG) techniques as needed to build sophisticated parametric models (Harries et al. 2015a). This also supports the conversion of geometric data from one format to another, feeding different tools with their required inputs. CAESES' heritage being simulation-driven design on the basis of computational fluid dynamics (CFD), the system also allows the generation of watertight tri-meshes as needed for grid generators, see Harries (2014) for an overview and Albert et al. (2016) for a detailed example.

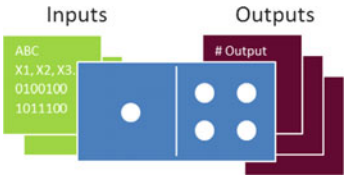
Finally, CAESES® offers a comprehensive feature technology to script additional analyses, for instance, a comparison of required versus attained EEDI, and to encapsulate higher-level objects, for instance, a Wageningen B-series propeller as elaborated in Harries et al. (2018). Features can access all objects available within CAESES®. They are interpreted code and can be reused, but also adapted, in different projects. Features can contain (internal) optimizations and even be nested. For convenient feature development, CAESES® allows both the interactive creation from chosen (geometric) models and writing code line by line while offering compiler functionality such as auto-completion of entities, error checks, and break points.

8.2.4 Integration Approach Taken in HOLISHIP on the Basis of CAESES

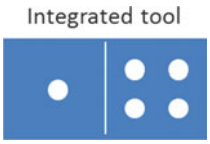
Practically all tools for analysis and simulation can be run in batch mode. Often they support setting up calculations interactively within a dedicated graphical user interface (GUI). Typically, all input data needed are either readily stored or can be recorded in a set of input files, commonly comprising configurations and geometry. While the actual calculations take place and/or when they are finished, both intermediate and final results are stored in one or several output files.

Usually, these calculations can be repeated subsequently by running the tool in batch mode. Using the same input data, the very same output data are generated.

(A) Standard tool execution: Tool is run interactively (GUI) or in batch-mode



(B) Tool can be run in batch-mode on the basis of interactive work (e.g. using recorded input)



(C) Integrated tool with input file(s) and output files(s) encapsulated (i.e., inputs and outputs are taken care of by CAESES®)

Fig. 8.2 Tool wrapping via its input and output files in CAESES®

However, if the input data are changed, for instance by providing a new geometry to be analyzed, different output data are produced. The input and output of data along with their encapsulation are shown in Fig. 8.2.

Within CAESES®, every tool that can be executed in batch mode can be readily coupled via its inputs and outputs. Any data item in the input file(s) can be tagged and, if wanted, replaced with a different value. Alternatively, a file can be replaced completely with a new version. This is often done for geometry files since very many data items change, e.g., all vertex positions of a NURBS representation in an iges file. Furthermore, any data item in an output file can be identified and read for further usage.

In order to establish the integration, all input file(s) and output file(s) are made known to CAESES®. They are used as templates, meaning that only a small number of data items are to be replaced and retrieved, respectively, while most data items are just kept as given, making the integrations light and flexible. Unaltered data items constitute input to and background information for the analysis and are considered constant for a particular design task. It should be noted that only those data items

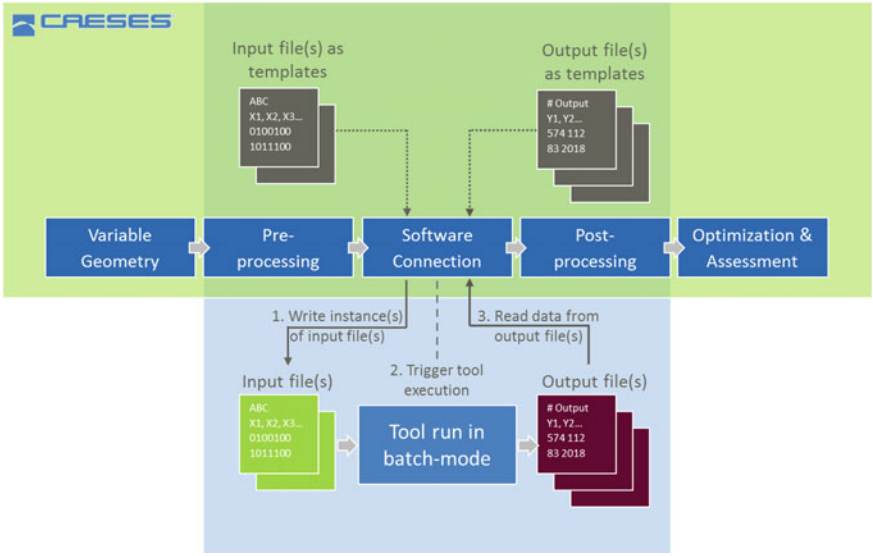


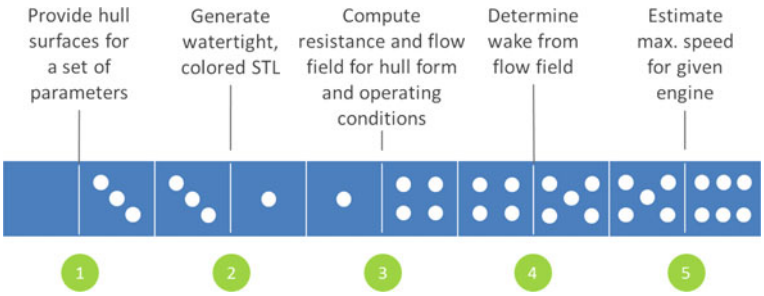
Fig. 8.3 Connecting tools within CAESES® via template files as part of a process chain

that shall be changed, that are needed for the design itself and/or that shall be passed on to another tool are managed. Figure 8.3 illustrates the usage of templates and the flow of data. See also Figs. 8.16a and Fig. 8.17 in Sect. 8.4 for data dependencies and data storage, respectively. Further details on how to set up a tool integration are given in Sect. 8.5.

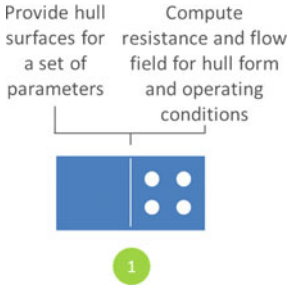
8.2.5 Encapsulating Tools

Integrating a tool in CAESES® via templates is not a very difficult or lengthy undertaking. Nevertheless, it naturally requires knowledge of both CAESES® and the tool to be coupled. A team of designers and engineers may be very interested in utilizing certain tools within their design work but would probably want to leave the specifics of how to set them up and how to make them accessible to others. Consequently, encapsulating tools for easy handling will increase the community of happy users that benefit from the caring experts. In this sense, an encapsulating tool can be interpreted as a technical APP or "domino" as suggested in Fig. 8.2c.

Depending on the complexity of the task that is encapsulated, a single domino may either solve just a small task or concatenate several tasks in order to realize a bigger "job." As an illustrating example, five dominos are shown in Fig. 8.4a which together shall provide an estimate of the speed attained for a given set of parameters that describe a hull form, an operational profile, and an engine. Clearly, every domino



(A) Several lower-level technical APPs (shown as dominos) concatenated to support a more complex simulation



(B) Higher-level technical APP combining parts of a complex simulation

Fig. 8.4 Technical APPs which wrap functionality for ease of use

in the sequence solves a small (sub) task. Several dominos, namely dominos 1–3 from Fig. 8.4a, could also be combined as depicted in Fig. 8.4b.

The provider of a domino (a technical APP) needs to compromise between ease of use and flexibility. Lower-level dominos may be more generic and can be further combined. Higher-level dominos may turn out to be more convenient if they offer the complete functionality as needed.

Furthermore, it is important to ask at what level a tool can or should be integrated. On one side of the spectrum, you can integrate a tool such that every possible command and workflow is supported within CAESES®. SHIPFLOW by Flowtech is a prominent example of such a tight integration, the actual GUI of SHIPFLOW being built on (a subset of) CAESES® (www.flowtech.se). On the other side of the spectrum, you can very loosely couple a tool if you only want to execute it for a clearly defined task, say, as part of a more comprehensive optimization. Somewhere in the middle of the spectrum are tool integrations that are task-specific yet generic enough so that a suitable range of analyses can be undertaken.

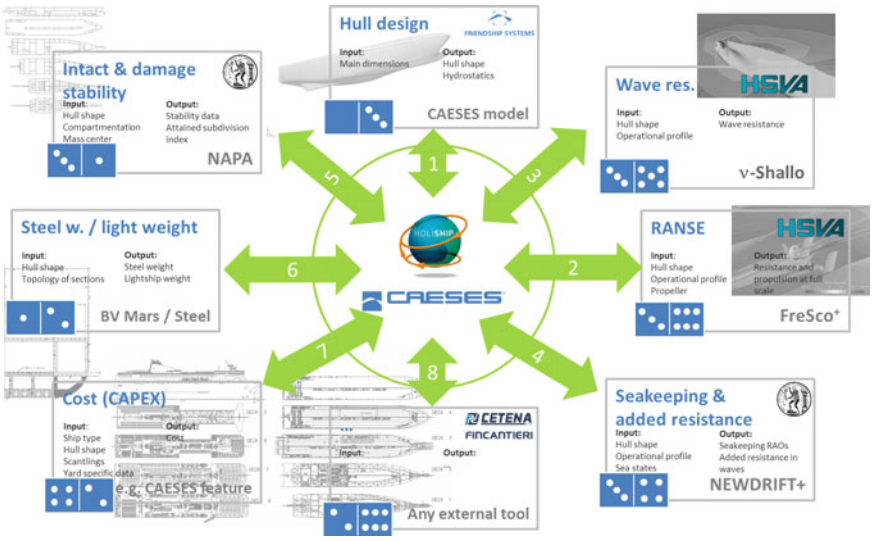


Fig. 8.5 HOLISHIP synthesis model based on CAESES® for the design and optimization of a RoPAX ferry

Within HOLISHIP, the integrations are chosen to be task-specific, see also Sect. 8.4. This is because any integration can start loosely and then grow step-by-step as the complexity demands. Very importantly, too, many tools are pretty powerful and are continuously developed further. Consequently, it would just not be practical to aim at an all-encompassing super system. Rather, the idea is to provide the functionality needed to address any new design challenge by bringing in additional tools and/or by extending the integration of tools that have already been coupled.

Figure 8.5 presents the synthesis model of a RoPAX ferry as an illustrating example of task-specific integrations, utilized for design and optimization (Harries et al. 2017). Several tools are combined so as to investigate the performance of the ferry with regard to resistance and propulsion, seakeeping, intact and damage stability along with estimates for weights, lane meters, costs, and the EEDI.

Specifically, the parametric CAD functionality within CAESES® is used to create a hull form (item 1 in Fig. 8.5). A watertight tri-mesh of the hull is then handed over to *FreSco*$^+$ by HSVA for viscous flow simulation of the baseline (item 2), using an stl file. This is followed by potential flow analyses of calm-water resistance for a large set of variants with *v*-Shallo by HSVA (item 3) for which CAESES® provides the discretized input via a dedicated panel file. Seakeeping behavior and added resistance in waves are then determined with NEWDRIFT+ by NTUA (item 4). Again, CAESES® delivers the necessary geometry input, here panels distributed up to the waterline written to a different panel file. For intact and damage stability, NAPA by NAPA Oy is run in batch mode (item 5). For this, CAESES® hands over the geometry of each variant as an iges file which NAPA imports and subsequently processes.

Additional analyses are made through external tools and features written within CAESES® that combine and/or post-process output from the various simulations (items 6 and 7). The extendibility of the synthesis model is illustrated by item 8.

In the synthesis model of Fig. 8.5, several dominos are shown, too. They serve to elucidate that various tools are combined at a certain level of abstraction, the number of points on either side of a domino giving an appreciation of input received and output delivered. Let us take the hull geometry from the parametric model as an example: The geometry is symbolized by three points, see also Fig. 8.4a. The geometry shows as output from the parametric model (item 1) and as input to the various simulation tools (items 2–4) even though different data and file formats are employed for the actual data transfer which is being taken care of by CAESES®. For geometry data that are used to feed the various tools, see also Fig. 8.18 in Sect. 8.4.

8.3 Variable Geometry

8.3.1 Geometric Modeling

In the design and optimization of maritime assets, geometry often plays a central role. Different to land-based facilities and plants, the geometry of ships and offshore platforms determines key performance aspects such as load-carrying capacity but also energy consumption, seakeeping behavior, comfort, and survivability. As a consequence, complex three-dimensionally curved shapes, featuring compound curvature, constitute the norm rather than the exception.

In principle, any CAD tool for geometric modeling that can be run in batch mode, even an instance of CAESES® itself, could be coupled to CAESES® so as to form part of the design synthesis model. However, since CAESES® already provides a comprehensive CAD functionality dedicated to the parametric modeling of hull forms, propulsors, appendages, etc., allowing the export of geometry data in both standard (e.g., iges , STEP files)[2] and tailored formats (e.g., panel and offset files), CAESES® can be utilized as both the platform for integration and the primary CAD engine.

For design and optimization, it is particularly important that geometry can be varied efficiently and at high quality. Efficiency relates to the effort needed to create a variant. Ideally, an update of geometry when producing a variant for design assessment takes only a few (split) seconds and not hours of manual work as would be the situation in a purely interactive modeling environment. High quality means

[2]The following standards are supported by CAESES®: Import formats: iges, iges (deprecated), SAT (ACIS), STEP, PARASOLID, stl, DXF (subset), Offsets (NAPA/SHIPFLOW), PFF (propeller free format); Export formats: iges, iges (deprecated), iges (STAR-CCM+), SAT (ACIS), STEP, STEP (STAR-CCM+), PARASOLID, TETIN, stl, stl (color), stl (multi body), stl (extract colors), stl (OpenFOAM), stl (STAR-CCM+), GridPro, Convergence, Wavefront (Obj), VTK Format, Offsets, Plot3D (panel mesh), PFF, GeomTurbo (NUMECA), DXF (subset), FSC (CAESES script).

that a small set of parameters already controls the geometry and that each variant produced is fair and potentially feasible so that any subsequent and time-consuming simulation becomes meaningful to launch.

There are two distinctive approaches which CAESES® supports both separately and in combination (yielding hybrid models):

- Fully parametric modeling (FPM) and
- Partially parametric modeling (PPM).

The former applies a hierarchical model built from scratch in which any variant constitutes an instance related to the chosen parameter values while the latter takes an existing CAD model and (only) modifies it parametrically.

The more powerful of the two approaches is fully parametric modeling since it may comprise and combine mathematical expressions, if-then clauses, cascading dependencies, all possible curve and surface entities, internal optimizations (e.g., to capture equality constraints), Boolean operations, etc. It is typically more demanding and time-consuming to build but can be applied at various stages, from early design to fine-tuning, by addressing different parameters within the same model. For example, at an early stage, the main dimensions may be subject to change. Later, as soon as the main dimensions are fixed, local parameters may be further adjusted, say parameters controlling the bulbous bow or the region around the propeller.

Partially parametric modeling is easier and faster to realize. An existing baseline, i.e., a CAD model that would often be called a "dead" geometry and that may stem from any CAD tool, is taken as a starting point. A number of transformations are subsequently imposed on the baseline, leading toward variants that feature new geometry for the initial topology. Scaling would be the most trivial yet very consequential PPM transformation. A prominent method with roots in animation and gaming is free-form deformation (FFD). In naval architecture, the Lackenby shift is a popular representative of PPM in which a concerted swinging of sections allows in- and decreasing as well as shifting of a vessel's displacement volume. Partially parametric modifications can be confined to certain parts of the geometry, and, importantly, within CAESES® several transformations can be concatenated. Yet, the essence and topology of the baseline always remain.

A detailed discussion of both fully parametric and partially parametric modeling along with further references can be found in Harries et al. (2015a). For further reading, see Harries (2010) which covers the FPM of a mega-yacht and MacPherson et al. (2016) which discusses the PPM of a patrol craft.

In the sections to come, two examples taken from the HOLISHIP project shall serve to illustrate FPM and PPM, along with hybrid modeling, so as to gain an appreciation of their role in design synthesis.

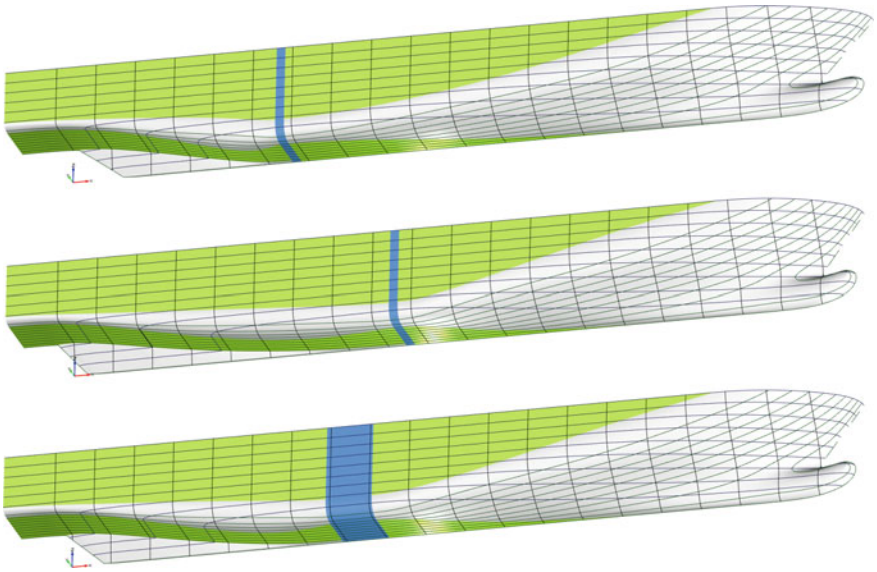


Fig. 8.6 Change in position and length of parallel mid-body

8.3.2 A RoPAX Ferry as an Example of Fully Parametric Modeling

A RoPAX ferry was used as a showcase for the integration of various tools as shown in Fig. 8.5 and discussed thoroughly in Harries et al. (2017) as well as in Marzi et al. (2018). A FPM was built within CAESES®, utilizing a design by Fincantieri as a reference.

The hull, stemming from a former R&D project, was made available as an iges file that contained numerous small surface patches. Instead of taking this CAD geometry as input to a PPM, a FPM was developed that followed the design idea without aiming at replicating the exact geometry.

To begin with, prominent shape characteristics were extracted from the initial design. All flat and developable surfaces were remodeled based on a dedicated CAESES® feature (as introduced in Sect. 8.2.3) that would identify the "rails" of such surfaces from the given hull. These curves were approximated using B-spline curves which were suitably parameterized. The main dimensions of the hull were introduced as global parameters, linking all control points to the overall size of the hull. Furthermore, relative measures were introduced so that any change in main dimensions yields a new shape that is still feasible and fair.

Figure 8.6 illustrates different shapes that stem from a variation of the relative position of the parallel mid-ship and its length. The ruled surfaces shown in green can be seen to follow the change of the parallel mid-body marked in blue.

While the ruled surfaces require only two parametric rails each, the remaining hull surfaces are somewhat more complex in shape. The bilge region in the aft body as well as the surfaces in the fore body, both shown in silver-gray in Fig. 8.6, are modeled using so-called Meta Surfaces.

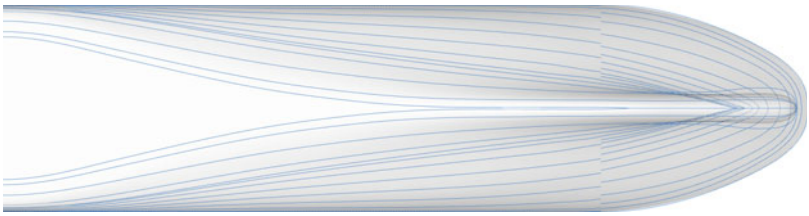
Meta Surfaces are powerful entities within CAESES®, specifically developed for the fully parametric modeling of three-dimensionally curved shapes. The idea of this type of surface is to sweep a certain building pattern, here a parametric section, along a dominant direction, here the hull's longitudinal axis. Importantly, the parameters that serve as input to the building pattern at each longitudinal position are not given as discrete values but are provided as parametrically controlled curves, too. This smoothly defines the gradual change as the section is swept, see also Harries et al. (2015a).

In the case of the RoPAX ferry, those (longitudinal) curves were again extracted from the given Fincantieri design, see Fig. 8.7a. Additional parameters were then introduced to achieve further variability to the model. Waterlines as derived from the Meta Surfaces are depicted in Fig. 8.7b, c for different parameter values. Differences in the resulting shapes can best be seen close to the flat-of-bottom.

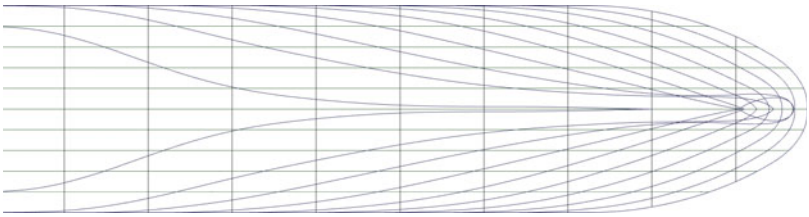
A substantial set of parameters make up the model in order to support a wide variety of global and local shape variations. Besides changing the main dimensions such as length, beam, height, design draft, position, and length of the parallel mid-body, it is also possible to modify shape details such as the bulbous bow's height, its length, volume, center of displacement, and inclination. Figure 8.8 shows a change of shape due to the modification of a single bulb parameter. Finally, a fully parametric skeg supports the setting of position, taper, etc., as illustrated in Fig. 8.9, while further parameters control the aft body.

The parametric rails of the ruled surfaces and the longitudinal curves used as input to the metasurfaces influence the hull shape directly while some parameters are not readily accessible yet that would typically be of interest to the naval architect. Primarily, that would be displacement volume and the longitudinal center of buoyancy. In principle, a sectional area curve could be utilized as input to the Meta Surface as shown by Harries (1998, 2010). Nevertheless, here a slightly more robust approach was taken: First, the "natural" hull shape that follows the rails and inputs to the Meta Surfaces is generated. Its associated displacement and center of buoyancy are then computed. If they do not meet the desired values, a Lackenby shift is undertaken, tweaking the hull shape to comply with the target displacement (or prismatic coefficient for given main dimensions) and center of buoyancy. Within CAESES® the Lackenby shift performs an internal optimization, actually hidden from the user, which ensures smooth transitions at either end when swinging the sections (Abt and Harries 2007b).

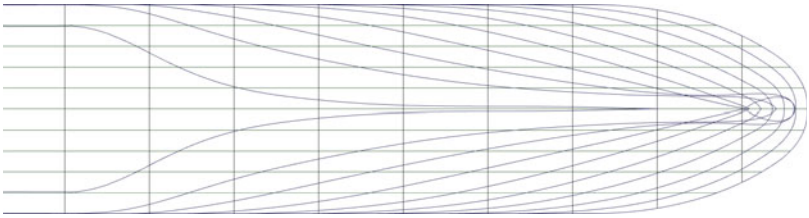
In addition, CAESES® allows further (nested) optimizations within a parametric model, purposely introduced by the user within a feature, in order to control certain parameters. For instance, in the RoPAX model at hand a one-dimensional search algorithm, here the so-called Brent, is employed to meet a given mid-ship coefficient. To this end, the bilge radius is used as an internal variable to attain the desired mid-ship coefficient while, within each iteration, the Lackenby shift is automatically



(A) Curves extracted from iges-file, serving as input to the Meta Surfaces



(B) Resulting surfaces with waterlines



(C) Resulting surfaces with waterlines for pronounced elongation of parallel mid-body

Fig. 8.7 Elements of a parametric model and resulting shapes

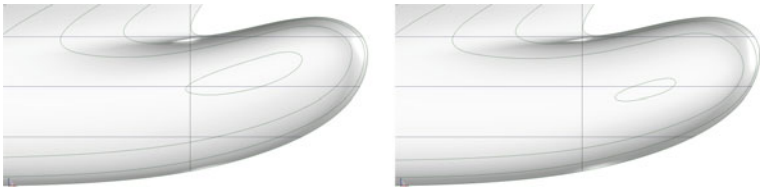


Fig. 8.8 Parametric modification of the center of displacement of the bulb

adjusted to ensure that the other settings are kept. Figure 8.10 depicts two hull forms for different mid-ship coefficients with all other parameters held constant.

The modeling techniques described above already constitute a very powerful FPM. Its user, be it a human being or an optimization algorithm, can address both global and local parameters with high efficacy. The direct control of global parameters allows the wide scanning of the design space (exploration) while local fine-tuning

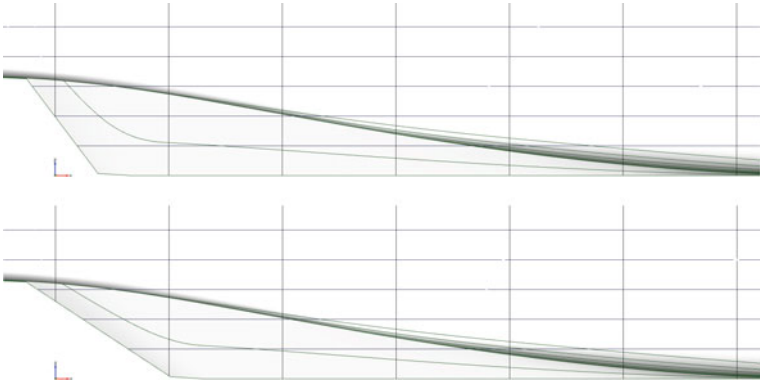
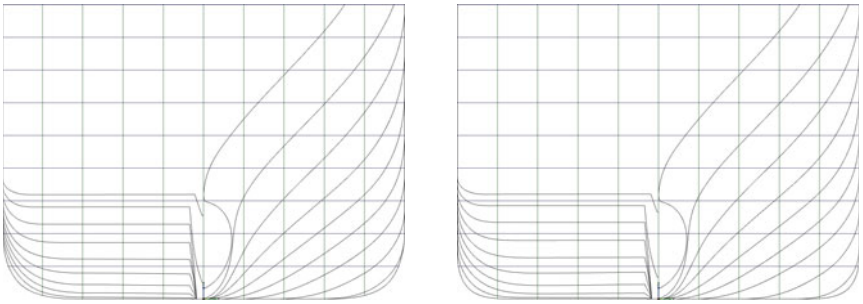


Fig. 8.9 Parametric modification of the skeg angle close to the transom



(A) Smaller mid-ship coefficient ($C_M = 0.965$), resulting in a larger bilge radius

(B) Larger mid-ship coefficient ($C_M = 0.985$), resulting in a smaller bilge radius

Fig. 8.10 Example of two different settings for the mid-ship coefficient with all other free variables kept constant

in critical areas such as the aft body and the bulbous bow are nicely supported, too (exploitation).

Finally, in the context of applying a parametric model within an automated optimization, it should be noted that the model may be built on dozens if not hundreds of parameters. Naturally, they are not all subject to change during a single optimization campaign. Rather, most parameters are set for a specific design task and then kept constant while only a few parameters, often not more than 10 to 20, are chosen for variation. These parameters are then treated as free variables, see also Sects. 8.4 and 8.6.

8.3.3 An OSV as an Example of Partially Parametric Modeling

PPM is often used in day-to-day work for quickly adapting a given geometry or conveniently investigating variants that do not deviate too much from an existing baseline. To this end, independent of the format in which a geometry has been imported (e.g., B-spline surfaces from an iges file or tri-meshes from an stl file), CAESES® offers a broad range of methods to evoke partially parametric changes to a geometry that was previously built within another CAD system (so-called dead geometry). The more prominent of the available PPM methods are

- Free-form deformation (FFD)
- Cartesian shifts
- Radial basis functions (RBF) and
- Morphing

For a detailed discussion of partially parametric modeling and further references, see Harries et al. (2015a).

In naval architecture typical modifications include scaling, change of both displacement and center of buoyancy as well as transformations confined to certain regions, for instance, modifying the bulbous bow to accommodate alterations in the operating profile. Within CAESES® a number of dedicated feature-based templates have been made available that provide the designer control over key parameters without the need of understanding the underlying mathematics.

A template project contains a predefined setup of features for analysis of an imported hull form and for the transformations acting on an image of the original shape. The process for generating a variant usually comprises four steps:

- Import of the baseline geometry from an iges file or similar
- Selection of the design draft for analysis
- Initialization, and finally
- Modification of the free variables.

Figure 8.11 depicts the hull form of an offshore supply vessel (OSV) for which such a template was used. Here, the baseline geometry came from NAPA and comprised a set of B-spline surfaces that were post-processed with CAESES® (e.g., to form a watertight definition, the deck being deliberately left out in the display). Important shape characteristics were identified and are shown in different colors.

It should be noted that the import of the baseline geometry is not a trivial step in the process. Exchanging mathematical representations of geometry, even though standards are utilized, bears the potential of errors due to misinterpretations between the sending and the receiving software tools. While B-spline curves and surfaces are exact representations of the shape, trimming operations are approximations[3] that depend on tolerances and the actual algorithms utilized. Boundary representations (BRep) form the basis of handling topological relationships between the geometric

[3] With the exception of iso-parametric trim curves in the domain of the parent surface.

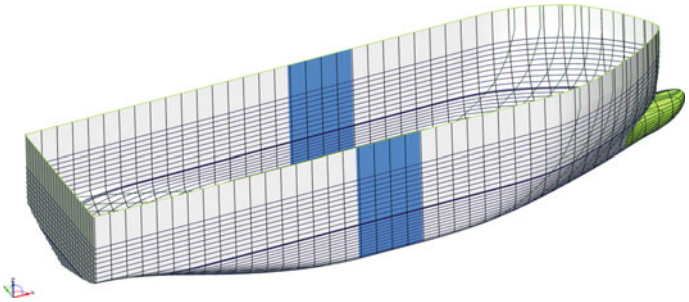


Fig. 8.11 Baseline imported from an iges file, comprising a set of B-spline surfaces

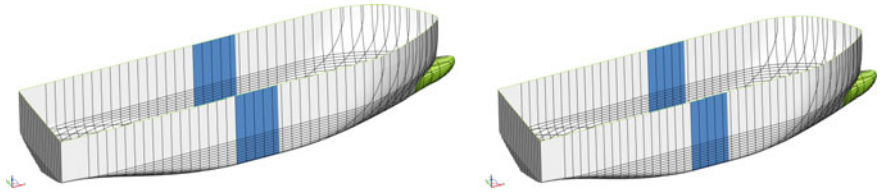


Fig. 8.12 Lengthened and shortened version of an imported OSV

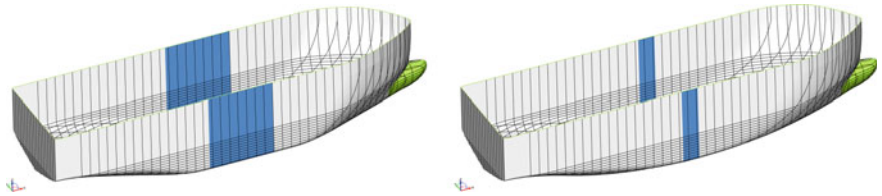


Fig. 8.13 Two hull forms with lengthened and shortened parallel mid-body (shown in blue) but identical displacement and longitudinal centers of buoyancy

entities, such as trimming and Boolean operations. The outcome of such operations, unfortunately, varies in different CAD systems. Sometimes deviations are small but occasionally operations cannot be successfully executed at all.

In hull form design shapes are often transferred as a pure patchwork of B-spline or NURBS surfaces (e.g., when using an iges export from NAPA) and sometimes as just one single surface (e.g., from MARIN's CAD tool GMS). CAESES® offers dedicated functions to repair and refine imported CAD models so as to prepare them for subsequent tasks.

In the example given, main dimensions are changed, see Fig. 8.12, and minor modifications of displacement volume and longitudinal center of buoyancy are undertaken, see Fig. 8.13. The latter two are realized by a Lackenby transformation, see Sects. 8.3.1 and 8.3.2, for a reference draft chosen by the user.

From the given input, baseline geometry and design draft, the variables are initialized, in particular, the main dimensions of the vessel. Complementary modifiers

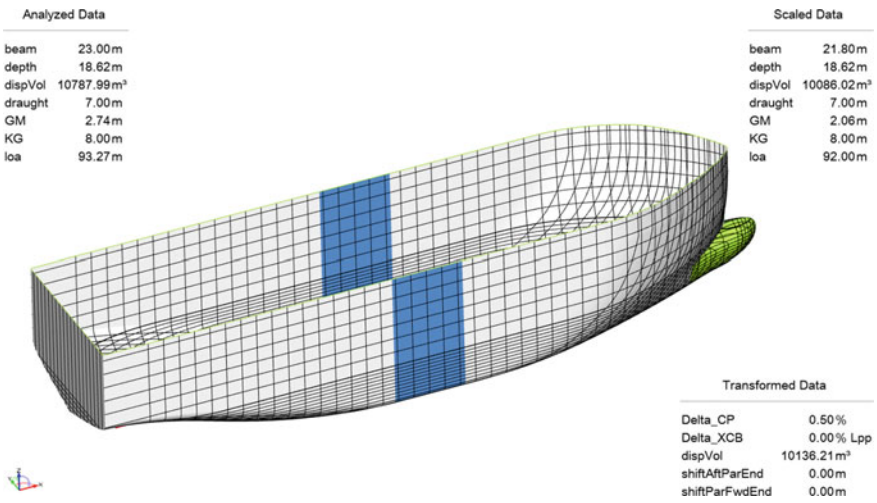


Fig. 8.14 Modified hull and control monitors for interactive shape adjustment

are set to zero at the start. These free variables allow controlling the bulbous bow and the location of the parallel mid-body. While Fig. 8.11 depicts the imported baseline, Fig. 8.14 shows a modified version along with the information displayed for interactive control of relevant parameters.

The partially parametric template discussed here forms the basis of a design space exploration in which various combinations of main dimensions were investigated. For given main dimensions, complementary parameters, in particular for the mid-body and the bulb, are (again) adjusted within an inner optimization loop. These nested optimizations are undertaken to come up with very competitive shapes for calm-water hydrodynamics for any current set of main dimensions. Further details are given by de Jongh et al. (2018). Figure 8.15 shows possible modifications of the bulbous bow shape.

8.4 Data Management

8.4.1 Hierarchical Models

CAESES® takes an object-oriented view toward modeling and data management (Abt and Harries 2007a). CAESES® objects can be asked for their attributes, values, and status, using *get* commands. Vice versa all objects can be assigned values, using *set* commands. The system allows introducing parameters and free variables without restraint. Parameters are mostly real or integer values that are either assigned a specific value, derived from other objects (e.g., output values from simulations) or computed

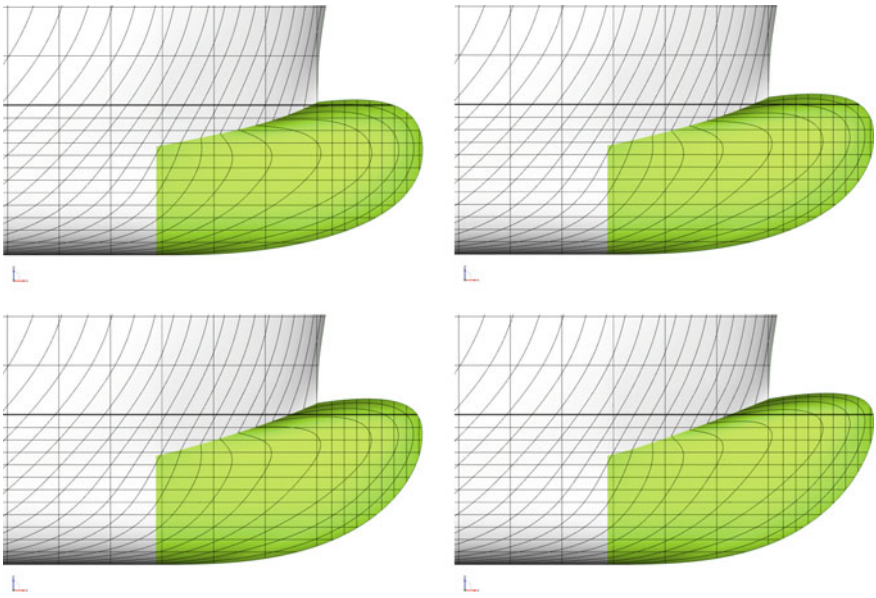


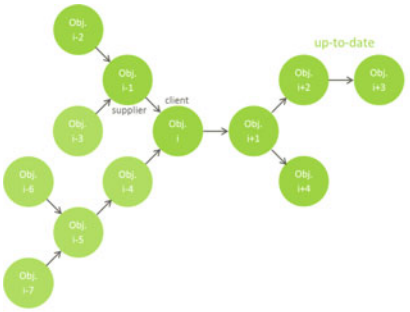
Fig. 8.15 Variation of bulb shape changing a single design variable

from mathematical expressions (often involving other parameters). Parameters form entries at the lowest level of a hierarchical model if they do not depend on any other object. A parameter at the lowest level of the model can be used as a free variable within an optimization, see also Sect. 8.4.2.

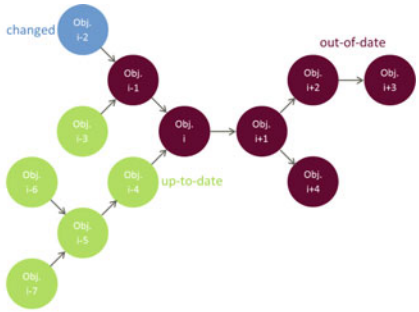
Figure 8.16 illustrates a hierarchical model (in abstract form) with a number of objects (here just 12 for simplicity). In principle, each object has one to several (lower level) suppliers and, vice versa, serves as a supplier to one or several (higher level) objects. When an object receives data, it is considered to be a "client." When it provides data, it acts as a "supplier."

In Fig. 8.16a, object i (called Obj. i) has two direct suppliers while it acts as supplier to just one following object (for simplicity), namely object i+1 (called Obj. i+1). Figure 8.16a shows a status in which all data are "up-to-date," meaning no changes have taken place since a complete update has been cascaded throughout the entire model. As soon as a single part of the model changes, all objects depending on it change their status to "out-of-date," see Fig. 8.16b. CAESES® would not necessarily update the model since, potentially, this may require considerable resources, for instance, if a CFD simulation is involved. Rather, the system waits for a request that an object needs to deliver data and then updates the model as need be. This so-called lazy evaluation (also known as lazy fetching) is depicted in Fig. 8.16c and d.

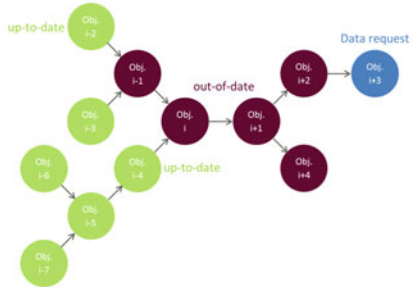
Objects and parameters can be flexibly introduced, erased, copied, renamed, and moved within the hierarchy as long as no circulate dependency is established. This



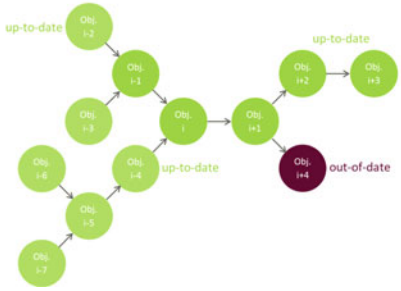
(A) Hierarchical model: All objects are up-to-date (indicated by objects shown in green)



(B) Objects that are out-of-date are not updated unless requested (here, objects that have gone out-of-date are shown in red while the one object that was changed is shown in blue)



(C) Object "Obj. i+3" (shown in blue) is requested to deliver data



(D) Object "Obj. i+3" is again up-to-date while "Obj. i+4" is still out-of-date since it has not been asked to deliver any data

Fig. 8.16 Lazy evaluation within CAESES®: Objects are only updated when they are needed (by another object or an action such as graphical display)

situation, however, is checked and excluded by CAESES® at the time of setting up a relationship between objects.

Importantly, an object may depend on an external simulation. This means that if the object is asked for its data, for instance, a parameter whose value is the resistance of a vessel at a specific draft and speed, it checks its current state. If it is up-to-date (e.g., none of the inputs to the flow simulation, in particular not the hull form itself, have changed), it can readily provide the requested piece(s) of information. However, if out-of-date (e.g., since a change in draft was made), it starts the necessary analysis. For the example of ship resistance, this may cause a full-fledged CFD simulation to be undertaken, unless deliberately suppressed, but may also just trigger the call of a surrogate model (see also Sect. 8.7).

8.4.2 Parameters Versus Free Variables

CAESES® allows defining parameters and free variables flexibly while working on a design and optimization project. Free variables within CAESES® differ from parameters as follows: A parameter can be any mathematical expression, a value taken from a simulation or just a number. Free variables, however, can only be numbers (or specific instances from a selection set). This is because they are controlled by the design engine during an optimization, i.e., they are varied by the strategy chosen for exploration (e.g., Sobol) or exploitation (e.g., MOGA). Hence, they need to be at the very start of any hierarchical model (as shown at the left end of Fig. 8.16).

Any parameter can be converted into a free variable and vice versa. In the latter case, this is straightforward since a free variable never has any suppliers to begin with. In the former case, however, it may cause the relationship to any suppliers, if given, to be cut since only the value itself can be associated with the free variable. This does not create any major problems but needs to be considered.

8.4.3 Bottom-Up Approach for Integration

It should be noted that the integration approach taken does not aim at holding all data items of a synthesis model that define a certain design task. Rather, tool specific data that neither are required to be shared nor are of interest to the design team are deliberately left out of the common database even though they form part of the wider repository.

For instance, a viscous flow code may allow setting a relaxation factor which an expert has tuned for a certain type of analysis. While this is certainly of importance to the CFD analysist, it may not be of major concern to the designer as long as the simulation converges and the quality of the results is ensured.

Figure 8.17 graphically depicts a synthesis model (in abstract form) in which three external tools are combined. Just a subset of all data is held within CAESES®. This subset will usually be larger than the sum of all individual intersections as illustrated by the white hexagon which comprises the data held centrally.

The reason for not sharing all possible data items is to support efficient—and possibly ad hoc—coupling of tools without having to first undertake lengthy data definitions. Naturally, there are advantages and drawbacks to this. An advantage of this so-called bottom-up approach is that the data to be shared can grow as the project advances. It has been found that defining data abstractly and completely ahead of a sophisticated project is not only time-consuming but also pretty difficult with many different tools and disciplines involved. A disadvantage of a bottom-up approach certainly is that databases cannot be readily recycled from one project to the next as

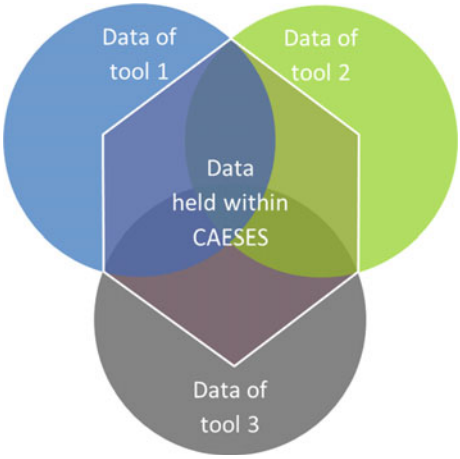


Fig. 8.17 Data storage within CAESES® illustrated for a setup with three external tools

that would ideally be possible in a comprehensive top-down approach.[4] Nevertheless, they can still be copied and, subsequently, amended and adapted.

8.4.4 Conversion and Enrichment of Data

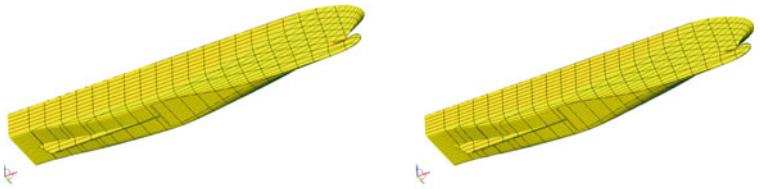
Data items from various tools can be brought together by means of expressions associated with a parameter or, if additional calculations or more comprehensive processing are needed, within CAESES' advanced feature technology, see Sect. 8.2.3.

When combining tools often the formats in which data are exchanged differ. For instance, hull forms sometimes need to be delivered as an iges files for one tool and as a watertight stl file for another tool while a third tool asks for a specific panel file or for a ASCII file with offset points, see Fig. 8.18. CAESES® may then act as a go-between, converting data to the format that suits each tool to be fed.

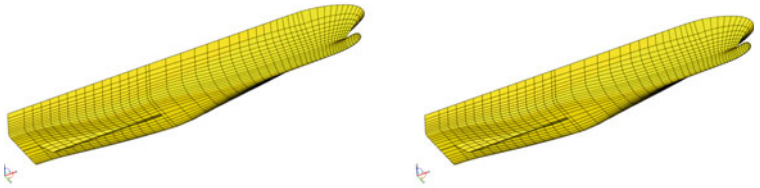
Figure 8.18 depicts various outputs for two different hulls, one being a long and slim variant and the other being a bit shorter and a little beamer design of the RoPAX ferry as introduced in Sect. 8.3.2. Figure 8.18a–e give the hulls' geometry, two different panel meshes, pure sectional data, and a wireframe model. Here, all these outputs form an integral part of the parametric model.

Furthermore, CAESES® can also be used to generate data that would otherwise be lacking. For instance, suppose that a baseline was available as offset data only. CAESES® could then be utilized to make the best fit of the points, yielding a set of fair curves that are subsequently skinned to provide a surface definition. From these enriched data new representations, say a watertight tri-mesh, would then be derived as needed.

[4]However, the vast range of maritime assets and different design scenarios would call for an extraordinarily large effort of defining a unifying data set.



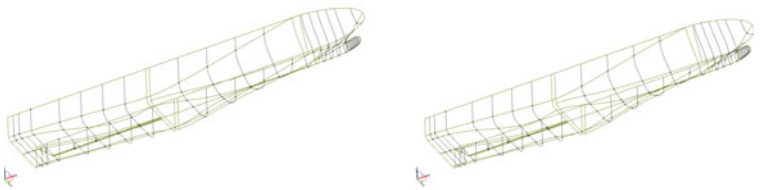
(A) Hull geometry for two variants (long and slender vs. shorter and beamer)



(B) Panel meshes for calm-water resistance analysis (input to ν-Shallo)

(C) Panel meshes for seakeeping analysis (input to NEWDRIFT+)

(D) Sectional data for structural analysis (input to MARS/STEEL)



(E) Wireframe model of surface patches for safety assessment (input to NAPA)

Fig. 8.18 Different data sets used to feed various simulation tools (shown for two instances of a RoPAX ferry)

8.5 Software Connection

8.5.1 Software Connector

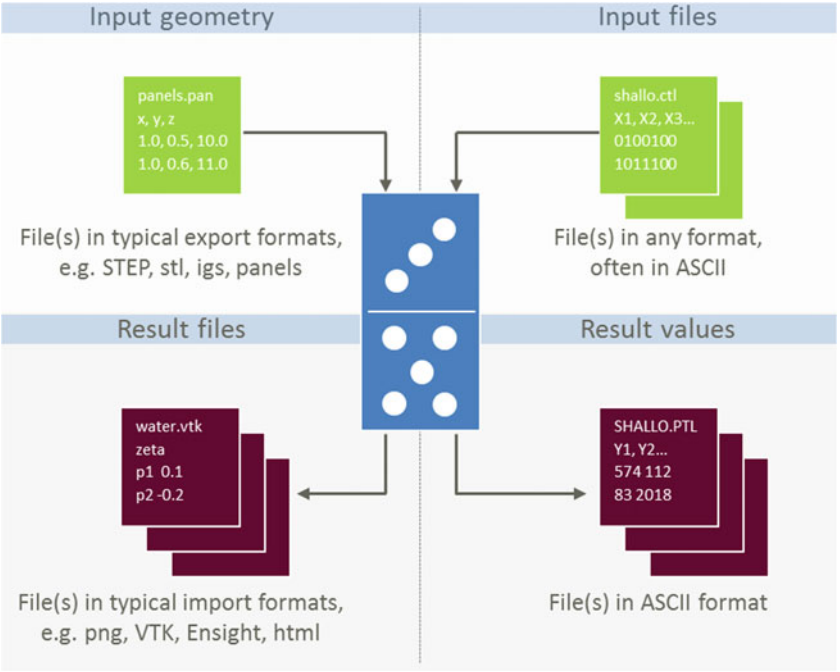
In order to prepare a tool to be connected to CAESES® a person familiar with the tool would first run it as a stand-alone application. He or she would provide all input and output file(s) that a batch mode execution requires, see also Fig. 8.3 of Sect. 8.2.4. These files are then loaded into CAESES® and used as templates as described in Sect. 8.2.5.

Figure 8.19 shows the so-called SoftwareConnector within CAESES® in both abstract and concrete form, using HSVA's wave resistance code ν-Shallo as an example. The SoftwareConnector provides four quadrants, two of which are for different types of input files while two of which are for different types of output files, supporting both binary and ASCII formats.

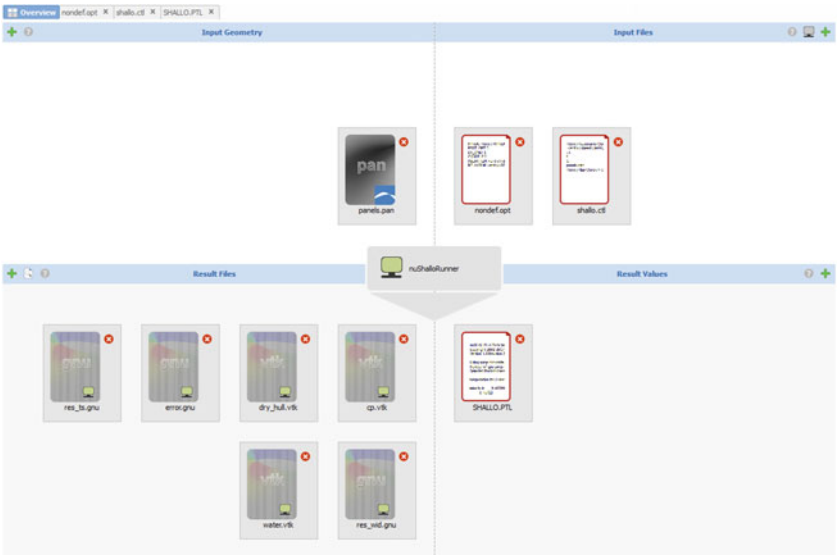
Certain types of input files can be handled directly, namely those which relate to the export formats readily supported by CAESES®, e.g., STEP, stl, igs, panel meshes. Likewise, for certain types of files that are output from the tool to be integrated, CAESES® provides interpreters, e.g., png, VTK, Ensight, Tecplot, and html. Furthermore, any data item in an input file can be tagged to be replaced by a parameter or free variable from the entire synthesis model. Similarly, any output file in ASCII format can be parsed by CAESES®. For flexibility, all sorts of identifiers can be utilized within a template, see also Sect. 8.2.4. Typical identifiers are characteristic names (so-called anchor strings), line and column numbers (also as offsets to anchor strings), relative position of a data item within a file (e.g., first or last occurrence).

Within the SoftwareConnector also the path to the executable of the tool to be integrated is specified. A tool can be run either locally or as a remote computation on a different computer or HPC cluster, be it internally via a company's intranet or externally via the Internet, for instance in the Cloud (see Albert et al. 2016). CAESES® allows for command line arguments and logical constraints. The latter would suppress tool execution if important prerequisites have not been met, for example, the quality of the mesh falls below a certain threshold, and hence, a lengthy CFD simulation may not be worthwhile to run.

For a discussion about further ways of connecting tools to CAESES® see Abt et al. (2009).



(A) Abstract view of SoftwareConnector



(B) SoftwareConnector for ν-Shallo within CAESES®

Fig. 8.19 Tool integration via templates

8.5.2 *Integration of a Single Tool*

Figure 8.20 illustrates the extraction of data from a ν-Shallo output file. Once the flow solver is executed, an ASCII file, here SHALLO.PTL, is written. The CAESES® parser extracts several important simulation results from the file which are shown in the left lower part of Fig. 8.20. The highlighted row in the right part of Fig. 8.20 depicts that a certain anchor string is searched for, namely CR (for residual resistance coefficient), and the value next to it is read. Since the number of iterations needed for a flow simulation to converge would typically not be known beforehand here the last occurrence of the anchor string is searched for, see upper left part of Fig. 8.20, and a parameter, here nuShallo_CR, is assigned to the value found for CR. This way the results are made known to the platform and become available for further processing within the synthesis model.

For an elaborate treatment of single tool integration, see also Harries et al. (2015b) and MacPherson et al. (2016).

8.5.3 *Integration of Several Tools*

For each tool to be integrated, a dedicated SoftwareConnector has to be set up within CAESES®. This is frequently done ad hoc whenever a new tool comes into play. A

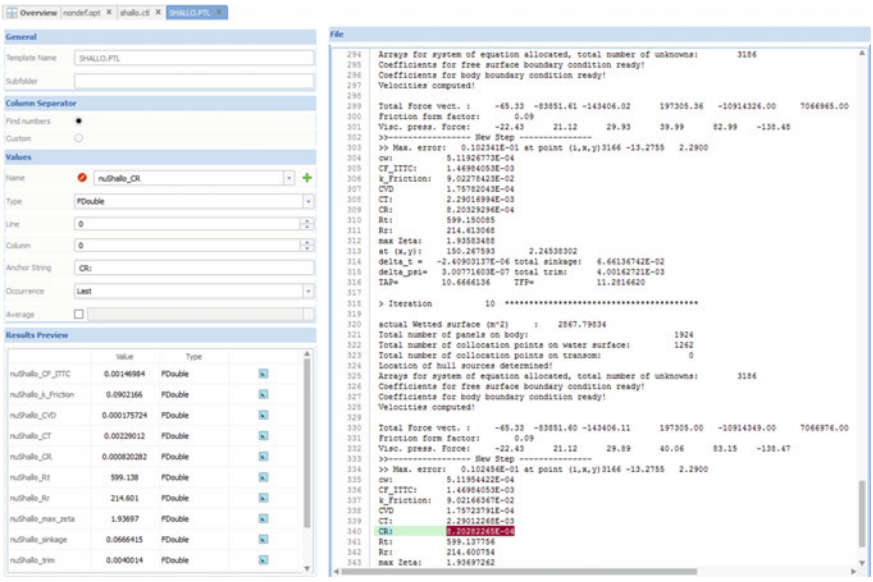


Fig. 8.20 Output file from ν-Shallo used as a template to identify data for extraction by the SoftwareConnector

SoftwareConnector, however, can also be imported from a previous project so that, quite often, only a few adjustments are called for.

Some tools are fairly sleek to connect while others may involve the handling of a lot of files. NAPA is an example for a tool that just needs a single script, and if hull variants are studied, an iges file for the current geometry. OpenFOAM is an example for a system that uses many input files, namely a watertight stl file for the geometry and quite a few control files. Still, the principle of integration stays the same independent of the number of input and output file(s).

In principle, a CAESES® project may comprise as many SoftwareConnectors as need be, readily allowing the flexible combination of several tools either to streamline a process that encompasses several tools to be run consecutively or to establish synthesis models of tools as introduced in Fig. 8.5. If the tools are independent of each other, they can be run in parallel by CAESES® (asynchronous update). Quite often, however, one tool requires input from another, and hence, a predefined sequence of execution has to be observed (synchronized update). This can be realized by making (parts of) the output of one tool (parts of) the input to another, i.e., by establishing a supplier-client relationship. As illustrated in Fig. 8.16, objects know the status they are in, namely up-to-date or out-of-date. If one tool, say tool A, requires input from another, say tool B, and that input is still up-to-date, then tool B does not have to be executed, but its data can be readily utilized. However, if tool B is out-of-date (or has not been run so far at all), then tool B would be triggered before tool A. If again tool B depends on yet another tool, say tool C, that would then be the first to be run, cascading through the hierarchical model until the first object is found that is up-to-date.

A further elaboration is given in Harries et al. (2017) regarding the synthesis model shown in Fig. 8.16 while an example of combining tools to establish a process chain for CFD, bringing mesh generation and flow analysis together is presented in Albert et al. (2016).

8.5.4 Connection with Other Frameworks

The integration of tools does not stop at connecting one or several stand-alone tools, but any other framework to which tools have been coupled can be connected as well. The prerequisites are that those additional frameworks themselves can be run in batch mode and that they allow setting parameters, launching encapsulated applications and providing result files. Since CAESES® can be easily run in batch mode and allows setting and getting of data, a straightforward scenario is to run a CAESES® project, a domino in the sense of Sect. 8.2.5, out of another CAESES® project. Nesting several CAESES® projects within one combining project would constitute a meta-project as further discussed in Sect. 8.9.1.

Alternatively, other frameworks, say a generic PIDO environment (like modeFRONTIER by Esteco or OPTIMUS by Noesis) or a CAE system (such as the ANSYS workbench), may be utilized to establish further levels of integration. A

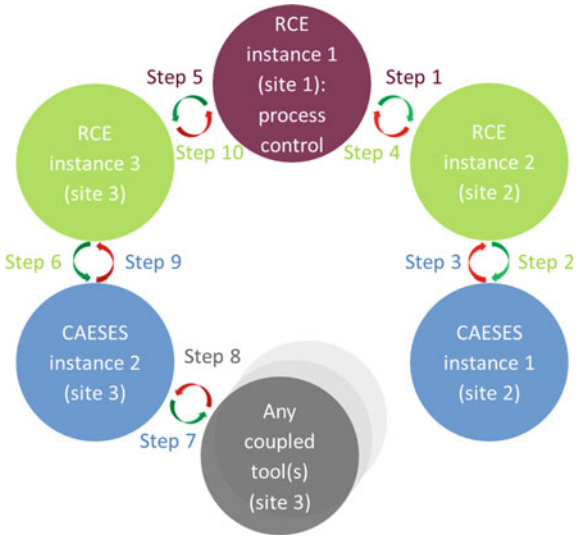


Fig. 8.21 Connection between RCE, CAESES® and additional tool(s)

pretty common application is that CAESES® acts as a pure geometry engine for parametric modeling. A more sophisticated scenario is that a framework calls one or several tools that are provided through CAESES®. This is illustrated in Fig. 8.21. Here the remote component environment (RCE) by DLR, the German Aerospace Center (DLR) being part of the HOLISHIP consortium, takes care of combining tools. Within the RCE, safe data transfer can be realized via the Internet between different sites and/or companies.

As shown in Fig. 8.21, a first RCE instance, being available at site 1 (e.g., in Hamburg, Germany) would take care of process control. It is connected with a second RCE instance that executes CAESES®, say as a geometry engine. This could be made available at any party connected to the Internet, say at site 2 (e.g., in Berlin/Potsdam, Germany). The first RCE instance gathers data from the second RCE instance and then transfers these data to a third RCE instance at yet another site (e.g., in Wageningen, The Netherlands). That third RCE could be connected to another instance of CAESES® which would make available additional tools already embedded. The sequence of data transfer is illustrated by the numbers associated with each step shown in Fig. 8.21. Once data have been produced, processed, and moved between RCE instance 2 and RCE instance 3 (steps 1–10), RCE instance 1 could trigger a new sequence, starting again with step 1. Naturally, this simple setup can be extended to further tools which could be either integrated within the RCE environment directly or, again, indirectly via CAESES®. If suitably and steadily extended that could form the basis of a powerful engineering ecosystem related to the Internet of Things (IoT), see also Sect. 8.9.2.

8.6 Optimization

8.6.1 Overview

Once synthesis models are set up they can be utilized interactively by a designer or run automatically at large scale. A designer will most likely investigate just a handful of variants manually, be it to get an appreciation of system behavior or to check if all tools work together smoothly. Studies with hundreds and even thousands of variants are carried out within formalized optimization campaigns, mostly during the night, the weekends or, if time-consuming direct simulations are involved, even over the course of several days. Utilizing surrogate models, see Sect. 8.7, moves the effort of heavyset computations upfront, enabling a tremendous speed up for the actual optimization phase which can then be realized within a timeframe of minutes to hours.

Mathematically speaking, optimization is finding the extremum (minimum or maximum) for one or several objectives under a set of (inequality and equality) constraints. All elements in the synthesis model that are under control of the design team and which are consciously selected for modification constitute the set of free variables, see also Sect. 8.4.2. It is the design team that then decides on the lower and upper bounds of these free variables, i.e., the range within which each free variable is permitted to change.

If a multi-objective design problem is posed, as is the typical situation in engineering, there typically is no solitary optimum but a set of solutions that are non-dominated (Pareto frontier), i.e., for which no single objective can be further improved without impairing one or several of the other objectives. From all feasible solutions, in particular from the Pareto optimal solutions, the team finally suggests the most favorable design according to the client's preferences.

In general, two major types of optimization strategies can be distinguished (Harries 2014):

- Design space exploration and
- Exploitation

In general, optimization strategies aim at scanning the design space (exploration) or improving the objective(s) (exploitation) with as few costly evaluations (simulations) as possible. If many evaluations are affordable—for instance, by employing surrogate models as discussed in Sect. 8.7—some strategies (e.g., genetic algorithms as a hybrid between exploration and exploitation) try to establish not only a local but also a global view. For an in-depth discussion of strategies that have been successfully applied to the optimization of maritime assets see, for instance, Birk and Harries (2003).

Inherently, a multi-dimensional design space, often of order 10 regarding the number of free variable but sometimes even of order 100, requires the evaluation of a lot of variants to establish a thorough understanding. This means that very quickly many hundreds or even thousands of designs have to be generated and assessed.

Looking at so many variants might, at first glance, seem to be a rather brute force method. Nevertheless, besides finding better (and possibly even the best) solutions there are many important benefits to gain:

- Seeing cause and effect relationships (for example between free variables and objectives)
- Understanding trade-offs between opposing objectives (in a multi-objective design scenario)
- Settling on what should be selected as an objective and what ought to be treated as a constraint (which sometimes is far from trivial)
- Producing a tangible number of feasible variants (in particular for a heavily constrained design problem)
- Identifying constraints that are particularly difficult to meet (and which could possibly be relaxed)
- Getting an appreciation of the overall potential for improvements
- Gaining a feeling for the design task and, furthermore
- Mitigating the risk associated with taking a design decision.

It should be noted that the strategies of exploration and exploitation as discussed here are usually based on handling floating-point numbers. Usually, the topology of the product is kept constant during a single optimization run. Different configurations, for instance, a single-screw versus a twin-screw vessel or a direct-drive shaft line with a diesel engine versus a diesel electric Azimuth thruster, are usually treated consecutively within separate optimizations. As soon as many variants for a small number of different topologies have then been investigated, the various design clusters are compared from which the best topology and the most favorable overall design can finally be identified.

Naturally, there are limits to this practical approach, for example, if the number, material, and type of stiffeners should be optimized within a structural optimization. Discrete changes like these would formally be captured via Boolean, string and/or integer variables. Unfortunately, this quickly sparks a combinatorial explosion.

Therefore, as a work-around topology-defining variables are sometimes simply treated as floating-point numbers, too. A standard optimization strategy for real variables would then be employed in which a roundoff to the closest integer value takes place or a look-up table is employed that relates the real variables to discrete entries (Zeitz et al. 2014).

8.6.2 Exploration

Before starting to push for improvements, it is often advisable to develop an understanding of the design space, i.e., to answer, at least by and large, the following questions:

- Which of the free variables are particularly influential and which may not be so important after all and could possibly be left out of further investigations?

- Are there many isolated regions in which good designs are found (which would be an indication of many local extrema) or are the objectives and constraints quite smooth with respect to the free variables?
- Do better variants lie rather in the middle of the design space or toward the chosen bounds?
- Which of the constraints are often active, i.e., violated, and which are actually not creating any problems at all and could be taken out of the investigation?

The group of strategies that spread variants in the design space either systematically or randomly are called design of experiments (DoE), see Siebertz et al. (2010) for an elaboration. Only if the number of free variables n is small, a fully populated matrix of variants can be investigated since the grid scales with g^n, with g being the number of (regular) grid points in each direction of the n-dimensional design space. For example, three variants per axis ($g = 3$) in an eight-dimensional space ($n = 8$) already call for 6561 variants. Hence, there are more sophisticated exploration strategies which try to produce as much insight with as few variants as possible.[5]

A popular DoE is the so-called Sobol algorithm as developed by Sobol (1976). It is a quasi-random approach that mimics the behavior of people at a beach: Unless they know each other, they tend to lie down at the spot that is furthest away from all other people so as to maintain utmost privacy. Every newly arriving person or group of persons would intuitively identify the region in which the free space is still the largest. Figure 8.22 illustrates the Sobol for a two-dimensional problem (first five variants).

The beauty of the Sobol procedure is that it produces the same spread in design space when repeated and that it can just be extended if further variants are desirable (Harries 2014). For instance, if the above questions cannot be answered reasonably well by the initially chosen number of variants, it is straightforward to complement the exploration with further variants that readily blend in. Finally, explorations like the Sobol are typically used to produce the data sets required to build surrogate models as explained in Sect. 8.7. The reason is that there is no unintended bias in the variants produced.

8.6.3 Exploitation

As soon as an exploration has led to a set of designs that, admittedly by chance, perform really well, a true optimization can be started. This means that a conscious push toward better solutions is made. To this end, the good or best designs found within a DoE, possibly along with the baseline, are so to say "exploited" by iteratively improving one or several objectives. There are numerous strategies to select from,

[5]The following strategies for exploration are provided internally by CAESES®: Sobol, exhaustive search, ensemble investigation, design assembler (externally defined matrix of variants), design lab (interactive variant creation). Furthermore, a range of complementary strategies are made available via DAKOTA by Sandia National Laboratories, e.g., Latin Hypercube and sensitivity analysis.

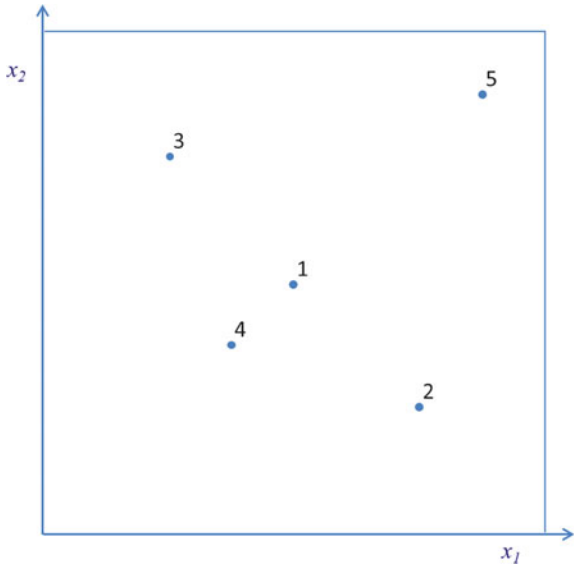


Fig. 8.22 Sobol sequence for five variants in a two-dimensional design space

and it is important to note that, unfortunately, there is no single strategy that would be best for all optimization problems. Thus, CAESES® offers a range of exploitation strategies.[6] Some strategies search locally, mostly (deterministic) search strategies, while others act more globally, like genetic algorithms, adding robustness albeit at the cost of considerable effort. Details and references are given in Birk and Harries (2003).

The principle idea of a deterministic search strategy is that of a blindfolded person in the mountains trying to find the nearest valley (minimization). The person would probe into the first direction, using a meaningful delta of the first free variable. If an improvement was found, a step forward would be made. Otherwise the opposite direction would be taken unless there is no improvement found there either. This is followed by a similar step into the second direction of the design space and so on. The search typically ends in a local minimum (or close to it). It is fairly quick but depends on the starting point(s). Inequality constraints are often handled during a search by means of an external penalty functions (sometimes by means of a barrier) unless a strategy already comes with an internal mechanism (e.g., the T-Search). The idea is rather simple: Whenever a constraint is violated the objective(s) are artificially worsened so that the search has no incentive to remain in the infeasible domain (or to leave the feasible domain in the first place). There are, not surprisingly, many search strategies around, some of which use objective values only while others make use of gradient information, too.

[6]The following strategies for exploitation are provided internally by CAESES®: Nelder–Mead Simplex, T-Search, Newton–Raphson, Brent (1d), NSGA II, MOSA. Furthermore, a large range of advanced strategies are made available via DAKOTA by Sandia National Laboratories, e.g., local optimization (multi-start), global optimization on response surface.

The principle idea of a genetic algorithm is that the objective(s) are interpreted as a fitness measure that determines an individual's chance of survival and of having children. A genetic algorithm commences with an initial population, possibly taken from a preceding DoE. The best individuals of that generation are selected to produce children. Similar to nature's standard approach, two individuals are paired, leading to a swap of parts of their free variables (their DNA) and randomly receiving (smaller or larger) mutations. This creates new individuals that belong to the next generation. The objective(s) and constraints are computed for this new generation, and infeasible individuals (designs) would be excluded as eligible parents. The best candidates are again selected for reproduction, leading to the third generation and so forth. Again, many different strategies and variations to the theme have been proposed and are in use.

8.6.4 Assessments

Once an exploration and/or exploitation has been run, CAESES® as the HOLISHIP platform for design and optimization allows scanning through the results by means table views, correlation diagrams, and a design viewer for direct comparison of variants. Figures 8.23, 8.24, and 8.25 show some of these assessment tools for the RoPAX ferry as discussed more thoroughly by Harries et al. (2017) and Marzi et al. (2018).

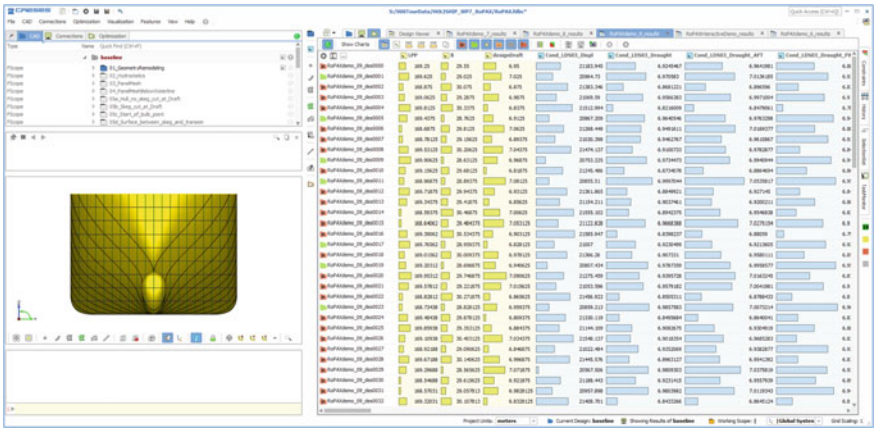


Fig. 8.23 Design table for free variables, objectives constraints, and further parameters of interest

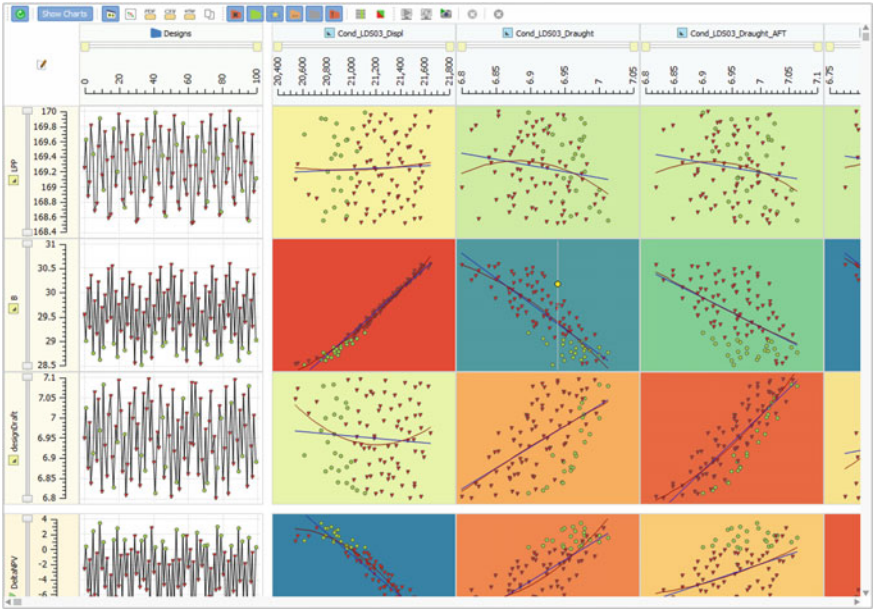


Fig. 8.24 Charts correlating objective and constraints with free variables

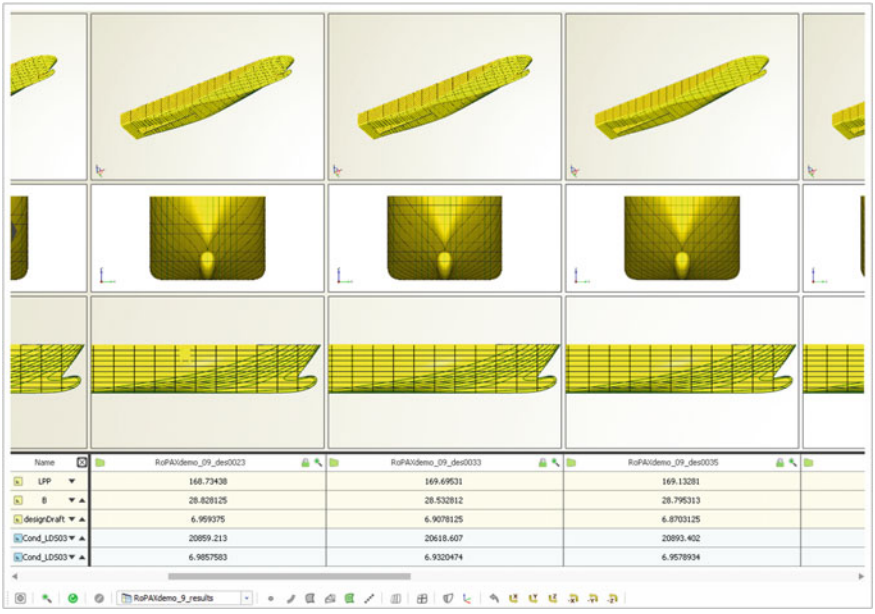


Fig. 8.25 CAESES® design viewer for graphical comparison of variants (excerpt from hundreds of variants)

8.7 Direct Simulation Versus Surrogate Models

8.7.1 Idea of Surrogate Modeling

Simulations vary in effort and resources from just a few CPU seconds to hours and even days of number crunching. Frequently, they require expert knowledge that is neither available at all times nor in every design team. Furthermore, resource-intensive simulations such as viscous CFD and damage stability assessment generally require special hardware, possibly even an HPC environment, and associated licenses.

A possible solution to this predicament is to replace direct simulations by so-called surrogate models, also known as meta-models or response surfaces. The concept is as follows: A sufficient set of simulations is undertaken upfront before the actual design work takes place. These simulations are then utilized to build an approximation, the surrogate model, which later on replaces the direct simulations within an acceptable level of accuracy. As can be readily seen, the effort of undertaking direct simulations cannot be avoided. However, it is shifted in time and to the right people.

Naturally, the variants for which the direct simulations are to be performed need to relate to the free variables that shall be subsequently used in the design task and/or the optimization campaign. Otherwise, the surrogate model will not depend on these free variables and cannot be utilized meaningfully. As a consequence, the important questions to ask are if it is actually required to engage in surrogate modeling and, if yes, how to do it properly. Table 8.1 gives some recommendations.

Table 8.1 Recommendations for replacing direct simulation with surrogate models

Resources needed for analysis per variant	Smooth behavior of response to be captured	Licenses and/or tool know-how locally available	Direct simulation	Surrogate model
Low	Yes	Yes	+	−
	Yes	No	−	++
	No	Required	++	−
Medium	Yes	Yes	+	+
	Yes	No	−	++
	No	Required	++	−
Large	Yes	Yes	+	++
	Yes	No	−	++
	No	Required	+	−

Low Less than one minute on a standard computer (PC or workstation)
Medium A few minutes to half an hour
Large Several hours to several days (possibly on a cluster or an HPC)
− Unsuitable
+ Recommended
++ Highly recommended

8.7.2 *Typical Surrogate Models*

The easiest surrogate model is that of a linear regression of a data set that depends on only one free variable as illustrated in Fig. 8.26. This is often done for experimental and empirical data. A line is fitted through the data points such that the square sum of the errors between the actual values and their approximations is minimized. Quadratic and even higher-order polynomials are also regularly used, in one-dimensional but also in n-dimensional regressions, n again denoting the number of free variables.

The input required to produce a meaningful surrogate model quickly scales up with n. The number of coefficients L needed for a (fully populated) n-linear response surface is $L = ½ (n \cdot (n + 1)) + 1$. Hence, as a rule of thumb, about n^2 independent data points are required if the response to be captured is well behaved and shall be replaced with more than just an n-linear model. Well behaved means that the response is rather smooth, i.e., that it does not oscillate (or jump) with regard to any of the free variables.

As can be readily imagined the number of free variables has to be sufficiently small if the simulations take a lot of resources. For example, if you need one hour of CPU time for the simulation of every variant, ten free variables already call for 100 h (more than four days) of number crunching. Cutting down to five free variables means that about 25 variants could be sufficient which could then be processed within a little more than a day.

The distribution of variants in the n-dimensional design space is clearly of importance, too. If the variants from which to derive the surrogate model were unfavorably placed, say all of them were aligned with regard to one of the free variables, clearly the model would not be able to capture any other dependencies. Two strategies of design space exploration are often employed to spread the variants in design space intelligently, the Sobol and the Latin Hypercube as elaborated in Sect. 8.6.1 on design of experiments.

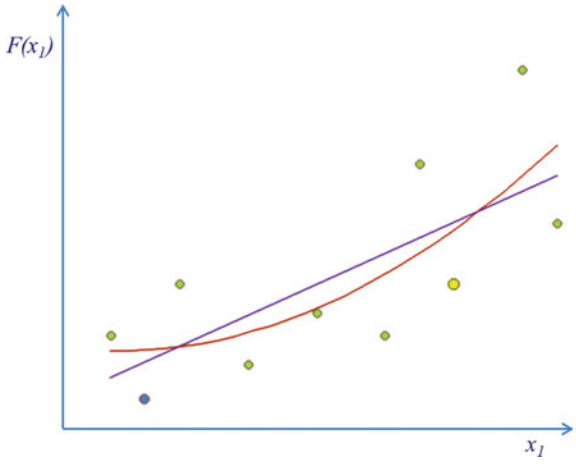


Fig. 8.26 Linear (shown in blue) and quadratic (shown in red) surrogate models for a set of data points

Popular surrogate models that are more sophisticated than polynomial regressions are artificial neural networks (ANN), Kriging and hybrid models that combine various surrogate models to increase the coefficient of prognosis, i.e., the accuracy of predicting system behavior.

ANNs are particularly suitable for responses that are not necessarily smooth and for which large data sets are available. In principle, ANNs mimic the human brain that, if fed continuously with input of a certain kind, learns to predict system behavior by suitably connecting neurons. For example, a goalkeeper after hundreds of hours of parrying practice has learned to predict the trajectory of an approaching ball. In many situations the prediction is good enough, leading to a save, while sometimes something unforeseen happens, causing a slightly different system behavior and, eventually, a goal. In order to build an ANN, the data set of the response to be captured is subdivided into a training set and an independent set for validation. For a thorough introduction to ANNs, used in maritime applications, see for instance Mesbahi in Birk and Harries (2003).

Kriging is a method that interpolates the training set while weighting the responses of various data points to predict system behavior at intermediate points. It was introduced by Danie G. Krig, a South African mining engineer, who wanted to predict the most likely distribution of gold based on samples from just a few boreholes, see for instance Press et al. (2007).

CAESES® supports quite many surrogate models by means of the DAKOTA software package from Sandia National Laboratories. Which of the surrogate models yields the best behavior and accuracy for a particular application is subject to testing. This does not create any major bottlenecks since the database for both training and validation can be run once and then utilized time and again.

For further elaborations see Myers and Montgomery (2009).

8.7.3 Using Surrogate Models

Once a surrogate model has been tested and found to be sufficiently accurate and reliable, it can be employed to replace the actual simulation. The advantage of speed is tremendous since a surrogate model would typically yield a result within seconds (if not split seconds) instead of minutes, hours, or even days of simulation time.

Figure 8.27 shows the replacement of several direct simulations within the synthesis model of the RoPAX ferry as given in Fig. 8.5 and discussed by Harries et al. (2017). Specifically, the wave resistance and seakeeping analyses by means of potential flow theory were replaced along with the laborious computations of damage stability. These types of analyses would typically require several minutes per variant, adding up to 15–30 min of overall simulation time for each design of interest. With the heavy number crunching replaced, however, the investigation only

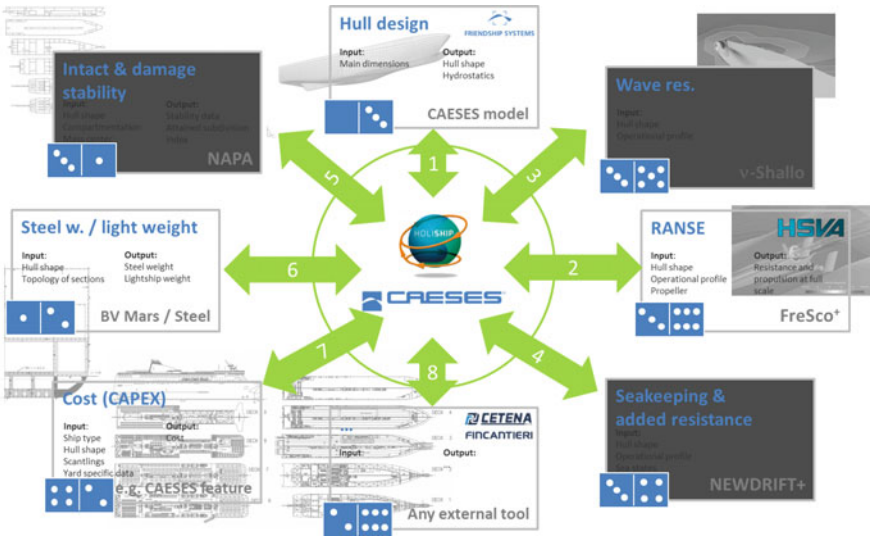


Fig. 8.27 HOLISHIP synthesis model based on CAESES® including surrogate models for the design and optimization of a RoPAX ferry

takes a fraction of a minute.[7] Hundreds of designs can thus be run within one to two hours, and thousands of designs can be studied over the course of a night (Harries et al. 2017). As a consequence, the designer gets a lot of insight into potentials and trade-offs along with an appreciation of dependencies and critical regions. Furthermore, it allows asking what-if questions and reacting to ad hoc modifications of the design requirements.

Figure 8.28 gives a comparison of the results from a surrogate model and data for the calm-water resistance prediction with ν-Shallo. The data from the surrogate model are plotted over the actual simulation data. If the surrogate model was perfect, all points would line up on a straight line with unit slope. This is not the case, yet the accuracy lies within the ±1% error lines which would be within the repeat accuracy typically associated with model tests.[8]

[7]One may wonder why the much more cumbersome RANSE simulation was not replaced by a surrogate model in the example given. This is because for the sake of reducing the overall computational burden the calm-water performance of the RoPAX ferry was first computed with FreSco+ as a viscous free-surface flow solver by HSVA, yielding the total resistance and propulsive efficiency of the appended baseline. Subsequently, a potential flow analysis of the non-linear wave resistance of the baseline's bare hull was run with ν-Shallo, the non-linear potential flow code by HSVA. The performance of each variant was then determined by means of the difference between the baseline's wave resistance and each variant's wave resistance computed with the same panel code, utilizing a comparable discretization.

[8]This does not mean, as should be well noted, that the accuracy of the CFD simulation itself is within ±1% of experimental data.

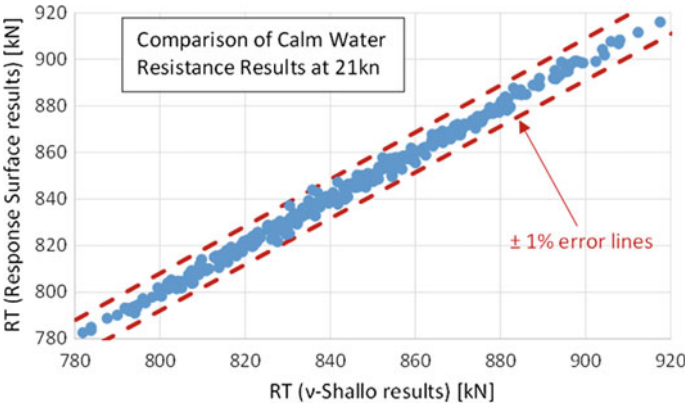


Fig. 8.28 Comparison between simulation data and surrogate model for calm-water resistance of a RoPAX ferry

Whether or not this accuracy level is sufficient needs to be decided on the basis of the design task for which the surrogate model shall be used. For finding the right dimensions of the ferry within a multi-disciplinary optimization, it seems to be perfectly acceptable while for fine-tuning a hull with respect to its hydrodynamics it may not yet be reliable enough. For further usage of surrogate models see, for instance, Harries (2010).

8.8 Scenarios of Application

8.8.1 Manual Versus Automated Design

In principle, CAESES®—and, hence, the HOLISHIP platform—supports different modes of operation, namely

- Manual (interactive) work
- Automated (formal) optimization and
- Batch mode execution.

Unless CAESES® is integrated itself, as discussed in Sect. 8.5.4, the two common scenarios are manual and automated design. Naturally, both manual and automated design work can be done at early stages, for instance, to identify main dimensions, and at later stages, for instance, to fine-tune the design to maximum energy efficiency.

When running an exploration or exploitation in an automated process, each variant that is produced receives a unique identifier under which all data associated with that design are stored. A folder for each design is created in the project directory, and all input and output file(s) along with additional files like screenshots are kept

unless explicitly tagged to be temporary. In this way, every variant can be studied afterward either within CAESES® or externally as if the design team had executed all simulations one by one and by hand. Hence, all data are open and accessible.

When doing manual design work, there are two convenient ways to use the system. Naturally, one can change settings and values for the baseline itself. This is quite common when building the hierarchical model and connecting to simulation codes. However, as soon as the synthesis model is established and a good starting point has been reached it may be of interest to keep that status as reference. In this situation, one can manually derive variants from the baseline for which all free variables and, possibly, any other parameter and/or relationship can be changed. The advantage is that the baseline is not touched, and hence, the design team is able to try out promising variants and use the baseline, or any of the variants, for comparisons. It should be noted that, in order to keep the storage requirements low, for each variant, only the changes to the baseline are stored within a CAESES® project. When doing automated optimization studies, the exploration and exploitation strategies discussed in Sect. 8.6 are put to use.

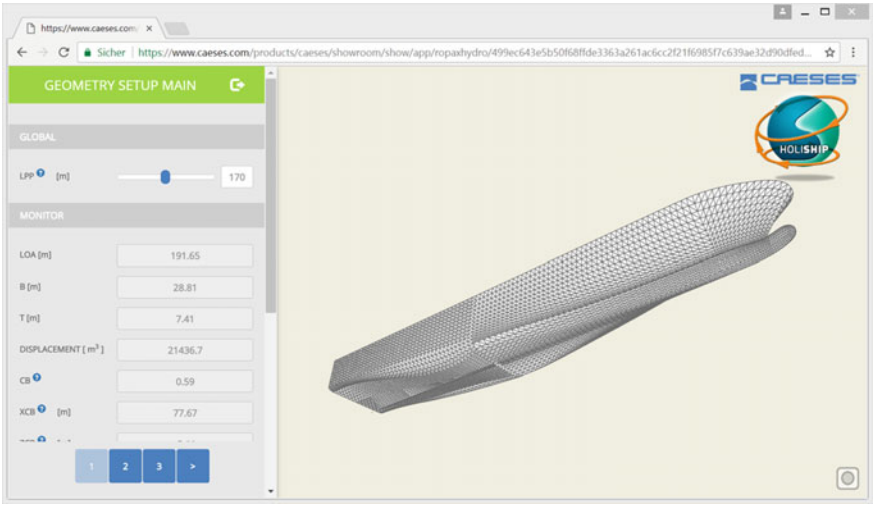
8.8.2 Offers via WebApps

In principle, any functionality that CAESES® offers and any tool that has been coupled to CAESES® can be made available as a technical application via the Web. These so-called WebApps, as discussed in Harries (2017) and Harries et al. (2018), enable access of sophisticated models and simulations through a Web browser. The aim of WebApps is to define meaningful workflows, configured in advance by an expert, that are easy to use by a designer. In that sense, a WebApp constitute a domino as introduced in Sect. 8.2.5.

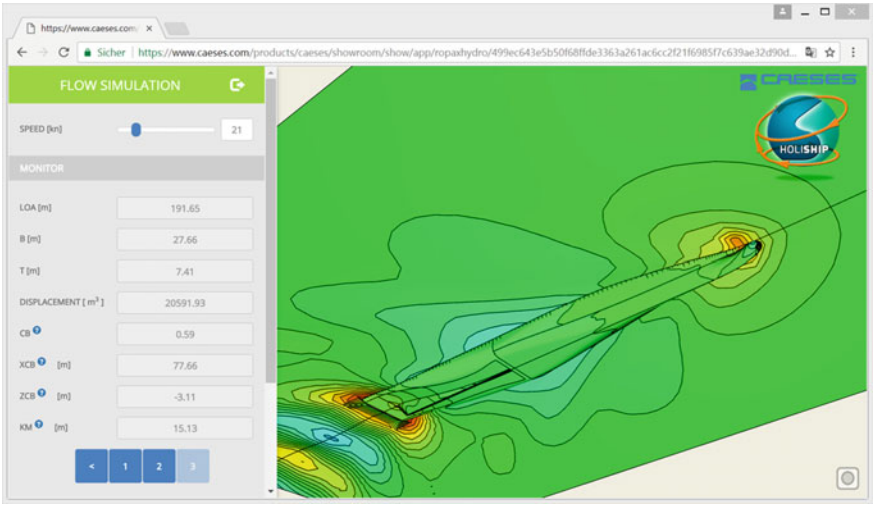
Naturally, the use case of a technical app is, by definition, somewhat confined. Essentially, the user shall be relieved from specialist knowledge associated with a certain tool but shall be put into the position of quickly producing reliable results for a specific task within given guard rails.

Of course, this is quite different to building and running synthesis models within a holistic approach that supports a range of workflows. However, the idea is that dedicated tools can be spread more widely and that, potentially, a future market place for a range of tools and associated services would help connecting designers, tool providers, and consultants. A number of technical apps could then form the basis for solving a more complex task.

Figure 8.29 illustrates a WebApp in which HSVA's non-linear potential flow code, ν-Shallo, is made available for analyzing a RoPAX ferry with regard to its wave resistance at various speeds. The topology of the hull—here a classic monohull as introduced in Sect. 8.3.1 with bulbous bow, transom stern, center skeg, and twin-screw arrangement—is static while the main dimensions and several local parameters can be changed within predefined bounds. Upon setting the accessible parameters as shown in Fig. 8.29a, CAESES® generates the geometry along with a suitable panel



(A) First page of the webApp, allowing the modification of the parametric model



(B) Third page of the webApp, allowing to choose vessel speed and launch ν-Shallo

Fig. 8.29 Resistance analysis of a RoPAX ferry via a WebApp (http://www.holiship.eu/approach)

mesh. As soon as the shape has been finalized a flow simulation can be triggered, yielding the resistance, the pressure distribution and the wave field as shown in Fig. 8.29b.

It should be noted that a standard browser acts as a front-end, replacing the detailed GUIs of CAESES® and the integrated code(s). For the Web-based application depicted in Fig. 8.29, the computations are run on a remote server. Alternatively,

a local setup can be provided, for instance, to assist less experienced members of staff with ready-to-go solutions prepared by a specialist within the team or by an external consultant.

8.9 Outlook

8.9.1 Meta-Projects

As can be imagined the complexity of projects that address the design of maritime assets holistically is high. For each simulation code that is integrated many parameters are involved, quite a few data items need to be provided and a lot of data are produced.

By using surrogate models the complexity can readily be reduced since a limited number of inputs typically lead to a small number of outputs as elaborated in Sect. 8.7. In addition, current development work for CAESES® aims at a plug-and-play approach in which projects can become part of an encompassing larger project, a so-called meta-project. Using one or several CAESES® instances in batch mode out of a CAESES® project that subsequently acts as the controlling platform is already possible as discussed in Sect. 8.5.4. Yet, an effort is made to further simplify the registration, execution, and data exchange between CAESES® projects, paving the path for providing, using and exchanging dominos conveniently and quickly.

8.9.2 Community of Providers, Consultants and Users

Taking a longer perspective, tool providers, consultants, and users are all believed to benefit from an open and flexible integration platform. Naturally, a certain market place would need to be created on top of the technical integration. While this is not part of the developments within HOLISHIP itself, the R&D project may serve to provide a critical mass of interested parties.

A market place would have to ensure that

- tools and services are made available
- can be booked and rendered as well as
- paid while
- intellectual property rights (IPR) are observed.

Furthermore, access rights and bidding techniques would have to be considered. This could democratize the access to tools, help the formation of temporary design teams across company boundaries and, as a consequence, lead to a growing network of stakeholders that may work together more swiftly and concurrently.

As an example, if a design team is interested in developing a new vessel for which they do not have reliable hydrodynamics data at their disposal, they could acquire

an existing surrogate model for the ship type in question, if available, or solicit for a consultant to build a model on their behalf. Similarly, consultants could anticipate a certain market demand and offer numerical hull series proactively. As another example, tool providers, be it companies or academic institutions, could offer their solutions more flexibly, reducing the threshold for non-expert users, and thereby increasing their group of potential clients and beneficiaries.

8.10 Conclusions

The European R&D project HOLISHIP addresses the integration and utilization of simulation tools from all disciplines relevant for the design of maritime assets. The tools come from different providers, use their own licensing schemes, and typically run either under Windows® or Linux™. Most tools have been developed over many years and are continuously worked on. In order to benefit from this vast pool of resources and expertise, a flexible and extendable integration approach has been favored. Consequently, CAESES® as a generic process integration and design optimization environment was chosen for the coupling of tools and the ad hoc definition of synthesis models.

A bottom-up approach is taken in which data are stored and exchanged only as actually needed for the design task at hand. Different to deploying a single legacy system this means that new tools and additional data can be quickly introduced and (re)combined, either by the tool providers or by the users themselves, as design tasks evolve and/or change. The integration requires that a tool can be run in batch mode. Input and output files are utilized as templates and only those data items that need to be varied, read, exchanged, or stored are addressed. Data that are used by just one single tool and that are not needed for design and optimization are not stored centrally but only kept and managed locally.

CAESES® acts as both a PIDO environment and a CAD system. The CAD technology within CAESES® aims at variable geometry for design and engineering as opposed to CAD for production. The system fully supports the definition of parametric models and of hierarchies. Dependencies between tools can be established so as to trigger automatic updates, ensuring consistency throughout both manual and automatic studies.

Within CAESES® a parametric model and a set of integrated tools constitute a synthesis model. Synthesis models are dedicated to scenarios within a certain scope such as the design of a twin-screw RoPAX ferry or the development of an offshore supply vessel for safe crane operations under DP. It can be run either interactively or automatically. Tools are triggered in parallel or, if one tool requires input from another tool, in the sequence of their dependencies. Furthermore, CAESES® supports the substitution of time-consuming and/or expensive simulations by means of surrogate models. Within acceptable accuracy levels, this enables a team of designers to run large investigations (several thousands of variants) both conveniently (even if not all tools are locally available) and quickly (within just a few hours).

Very importantly, CAESES® readily supports formal optimization campaigns. To this end, it provides a range of strategies for both exploration (e.g., Sobol) and exploitation (e.g., MOGA), triggers the execution of external simulation tools, collects results, manages the variants, and offers advanced methods of design assessment.

Acknowledgements We would like to thank Heinrich von Zadow, FRIENDSHIP SYSTEMS, for his support of the HOLISHIP project, his work on the parametric model of the RoPAX ferry and his contribution to this chapter.

References

Abt C, Harries S (2007a) A new approach to integration of CAD and CFD for naval architects. In: 6th international conference on computer applications and information technology in the maritime industries (COMPIT 2007), Cortona, Italy, Apr 2007

Abt C, Harries S (2007b) Hull variation and improvement using the generalised Lackenby method of the FRIENDSHIP-framework. The naval architect magazine, Issue Sep 2007

Abt C, Harries S, Wunderlich S, Zeitz B (2009) Flexible tool integration for simulation-driven design using XML, generic and COM interfaces. In: 8th international conference on computer applications and information technology in the maritime industries (COMPIT 2009), Budapest, Hungary, May 2009

Albert S, Harries S, Hildebrandt T, Reyer M (2016) Hyrodynamic optimization of a power boat in the cloud. In: High-performance marine vehicles (HIPER 2016), Cortona, Italy, Oct 2016

Birk L, Harries S (eds) (2003) OPTIMISTIC—optimization in marine design. WEGEMT summer school. Mensch & Buch Verlag, Berlin. ISBN 3-89820-514-2

Bostrom N (2014) Superintelligence—Paths, dangers, strategies. Oxford University Press, Oxford. ISBN 978-0-19-873983-8

de Jongh M, Olsen KE, Berg B, Jansen JE, Torben S, Abt C, Dimopoulos G, Zymaris A, Hassani V (2018) High-level demonstration of holistic design and optimisation process of offshore supply vessel. In: 13th international marine design conference (IMDC 2018), Helsinki, Finland, June 2018

Harries S (1998) Parametric design and hydrodynamic optimization of ship hull forms. Ph.D. thesis, Technical University Berlin, Mensch & Buch Verlag, ISBN 3-933346-24-X

Harries S (2010) Investigating multi-dimensional design spaces using first principle methods. In: Seventh international conference on high-performance marine vehicles (HIPER 2010), Melbourne, Florida, USA, Oct 2010

Harries S (2014) Practical shape optimization using CFD. Whitepaper. Retrieved from www.friendship-systems.com. Abridged version in the naval architect magazine. Issue Apr 2018, Nov 2014

Harries S (2017) Living in the cloud—How web-based apps will make high-tech more accessible. The Naval Architect Magazine. Issue Sep 2017

Harries S, Abt C, Brenner M (2015a) Upfront CAD—Parametric modeling techniques for shape optimization. In: International conference on evolutionary and deterministic methods for design, optimization and control with applications to industrial and societal problems (EUROGEN 2015), Glasgow, UK, Sept 2015

Harries S, MacPherson D, Edmonds A (2015b) Speed-power optimized AUV design by coupling CAESES and NavCad. In: 14th computer and IT applications in the maritime industries (COMPIT 2015), Ulrichshusen, Germany, May 2015

Harries S, Cau C, Marzi J, Kraus A, Papanikolaou A, Zaraphonitis G (2017) Software platform for the holistic design and optimisation of ships. 112. Hauptversammlung Schiffbautechnische Gesellschaft, Potsdam, Germany, STG, Bd. 111, Nov 2017

Harries S, Lorentz K, Palluch J, Praefke E (2018) Appification of propeller modeling and design via CAESES. In: 17th computer applications and information technology in the maritime industries (COMPIT 2018), Pavone, Italy, May 2018

MacPherson D, Harries S, Broenstrup S, Dudka J (2016) Real cost savings for a waterjet-driven patrol craft design using a CAESES-NavCad coupled solution. In: 15th computer applications and information technology in the maritime industries (COMPIT 2016), Lecce, Italy, May 2016

Marzi J, Papanikolaou A, Brunswig J, Corrignan P, Zaraphonitis G, Harries S (2018) HOLISTIC ship design optimisation. In: 13th international marine design conference (IMDC 2018), Helsinki, Finland, June 2018

Myers RH, Montgomery DC (2009) Response surface methodology: process and product optimization using designed experiments. Wiley, Hoboken

Press W, Teukolsky S, Vetterling W, Flannery B (2007) Numerical recipes—The art of scientific computing. Cambridge University Press, New York. ISBN 9780521880688

Schrage M. (2000) Serious play: how the world's best companies simulate to innovate. Harvard Business School Press, Boston, ISBN 0-87584-814-1

Siebertz K, van Bebber D, Hochkirchen T (2010) Statistische Versuchsplanung, design of experiments. Springer, Berlin, Heidelberg

Sobol IM (1976) Uniformly distributed sequences with an additional uniform property. Zh Vych Mat Mat Fiz 16:1332–1337 (in Russian); USSR Comput Math. Math Phys 16:236–242 (in English)

Walsh J (2018) Design space exploration—Markets and opportunities. Ora Research LLC and intrinSIM LLC, 224 pp

Zeitz B, Harries S, Matthiesen A, Flehmke A, Bertram V (2014) Structural optimization of midship sections for container vessels coupling POSEIDON with CAESES/ FRIENDSHIP-framework. In: 13th computer applications and information technology in the maritime industries (COMPIT 2014), Redworth, UK, May 2014

Stefan Harries Managing director of FRIENDSHIP SYSTEMS AG with the focus on Management, Strategies and R&D. Graduate of Naval Architecture from TU Berlin (Diplom-Ingenieur in 1992, Ph.D. in 1998). Holds a BSc in Mechanical Engineering (TU Darmstadt, 1988) and an MSE from the University of Michigan (1990). Associated with the Technical University Berlin as lecturer for Simulation-driven Design. His work is backed by experiences as scientist, hydrodynamicist and researcher in the design and operation of marine systems; published extensively on the parametric modeling and optimisation of ships.

Claus Abt Managing director of FRIENDSHIP SYSTEMS AG with focus on Products and Technologies. Studied naval architect (graduate of Technical University Berlin, Diplom-Ingenieur in 1998) with a specialization in Computer Aided Design and information technology. Simulation-driven design expert with more than twenty years of experience in engineering consultancy and R&D of product modeling, naval architecture and ocean engineering. Internationally, he has published on parametric modeling of free-form surfaces, software integration and numerical hull form optimisation.