SHIP CONCEPT DESIGN BASED ON A 3D-CAD-SYSTEM INCLUDING A REQUIREMENT VERIFICATION

H Lindner and **R Bronsart**, University of Rostock, Germany

SUMMARY

A system architecture allowing designers to create a ship concept design in 3D from the first sketches on is described in this paper. The centrepiece of the system architecture is a combination of a CAD system with well-developed modelling capabilities and a product data management system (PDM). This paper will show how the PDM system is extended in order to store and check functional requirements. A modular approach allows the integration of this system in the shipbuilding specific heterogeneous software infrastructure. The example will show how the different tasks in the concept design are supported by the system architecture including basic functionalities for requirement testing.

1. INTRODUCTION

Ship design is almost completely performed with the help of 3D CAD systems. However, the early ship design, especially at the pre-contractual stage, is often still developed and documented based on 2D drawing tools, because modelling and modification of a 3D model is a too complex and time consuming process. Therefore, the need for an efficient-to-use 3D CAD system combined with a PDM system in the concept design phase is apparent.

In order to apply a 3D modelling tool linked with a product data management system within the concept design, a novel system infrastructure consisting of interfaced specialized components has been realised in close cooperation with two shipyards. The system infrastructure is designed to suit the following key requirements:

- 3D-modelling: fast and flexible definition of the ships geometry
- Information-handling: capability of handling all relevant information
- Data-sharing: assuring that all parties involved in highly parallel design tasks work on the same data
- Requirement-tracking: store and associate design requirements with information objects

Requirement management is getting more important with the increasing complexity of the design. Customers more often request documented requirement breakdowns in some industrial sectors. Formalisation of the requirement definition can generate connections between the relationship model of the demands and the ship design model. Based on these links the concept design can be checked against the owners demands and the eligible statutory regulations.

A system infrastructure is presented, in which a general purpose 3D-modelling tool is linked with an external PDM system. The PDM system is the core-component of the infrastructure and acts as information hub for all related tasks. The geometry resides in the CAD system and is linked to the PDM system via a bidirectional interface for an automatic synchronisation of the data.
Based on an example ship it is shown that the implemented system architecture can be applied in the early ship design phase to develop a 3D ship model from the very beginning.

2. SYSTEM ARCHITECTURE FOR THE EARLY SHIP DESIGN STAGES

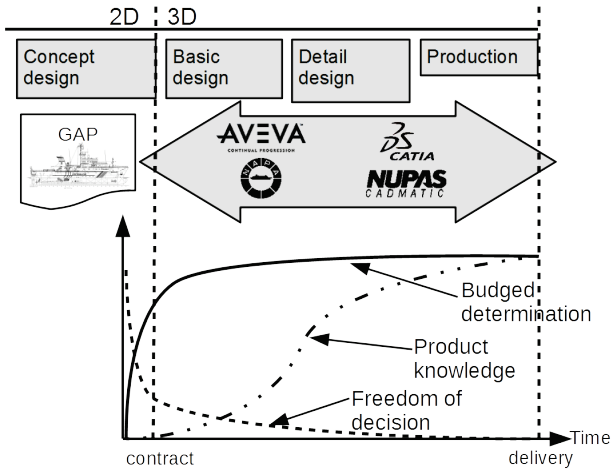


Figure 1: Budget determination, freedom of decision and product knowledge in different design stages. adopted from [1]

Ship design involves typical design stages such as concept design, basic design, detailed design and production. Shipyards usually deal with different important qualitative variables from concept design to production stages which are the following: Budget determination, product knowledge and freedom of decision. These variables alter over time as shown in Fig. 1 (adopted from [1]). In the beginning of the design process the design options are high and the budget constraints are low. Major decisions made in this first phase lead to a highly determined budget. In addition these decisions are made with limited product knowledge. The figure also denotes that most of the budget is determined in the very short time between request and

signed contract whilst the product knowledge stays low. This implies a high risk for the shipyard.

Fig. 1 also shows that the main CAD Model in the concept design is the general arrangement plan (GAP) which is generally a 2D drawing. However, many specialised CAD systems (AVEVA, NAPA etc.) are used in the concept design but not as main source of information as the GAP. After completion of the concept design and conclusion of contract, more sophisticated tools are used for different tasks such as the compartmentation or the steel design. These tools are based on 3D models derived from the general arrangement drawing and related documents. With these models and other further details the product knowledge rises, but along with the level of detail increases the freedom of decision decreases.

In summary, Fig. 1 shows that major decisions are made in spite of lack of detailed product knowledge. In order to reduce the technical and financial risk for the shipyard the improvement of the product knowledge especially in the concept design stage is crucial. In this paper a system architecture will be presented to replace the general arrangement drawing with a 3D model developed in a CAD system. Together with a PDM the product knowledge will increase significantly and therefore reduce the risk. To facilitate the process of reviewing the design against requirements the PDM was enhanced to store and check functional requirements.

2.1 GENERAL ARRANGEMENT – MAIN SOURCE OF INFORMATION

The general arrangement model is the main source of information in the concept design. It is common practice to develop and document the general arrangement with 2D drawing tools and associated spreadsheets. In this paper a system architecture is presented which replaces the general arrangement drawings with a 3D CAD model. Every individual task of the concept design process including the related information need to be determined to develop this system.

The concept design is an iterative process to refine the design in each iteration. In Fig. 2 the design spiral of the concept design is presented. This Figure is adopted from Eyres and Bruce [2].

In Fig.2 all tasks which are performed in the CAD system are coloured light grey. One of the first tasks is the subdivision of the ship into compartments and, later on, into tanks and rooms. In the next task the compartments receive attributes such as labelling and purpose. This kind of information is needed to check the design in subsequent tasks. Another major task performed in the CAD system is the arrangement of equipment. The steel design itself is not carried out in the CAD system but all information related to space requirements needs to be transferred to the general arrangement model.

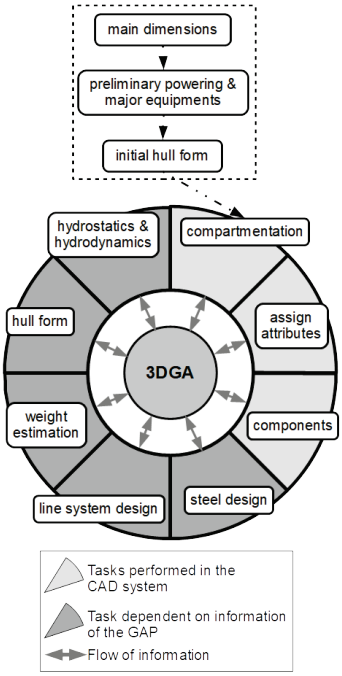


Figure 2: Design tasks in the concept design stage

Tasks relying on the information in the system are shown on the left hand side of Fig. 2. Mostly, these tasks check the design and are performed in specialised software tools

that need to be interfaced by the CAD system. The ship should not be completely remodelled in these tools to reduce the time needed for the whole design and the chances of errors. An important task is the development of the hull form including hydrostatic and hydrodynamic calculations. These tasks are performed in specialized software tools like NAPA, AVEVA, etc. Theses tasks rely on information from the 3D general arrangement (3DGA) but also giving back information to the 3DGA like the hull form, hydrostatic and hydrodynamic parameters.

2.2 SYSTEM REQUIREMENTS BASED ON PROCESS ANALYSIS

Based on the concept design process shown in Fig. 2 all tasks are fragmented to utmost simplicity in order to set up a system infrastructure which is tailored to the needs of the concept design. These functions were structured in trees with four main branches shown in Fig. 3. The branches contain functions with the following types:

- 3D-modelling capabilities: The time needed for the concept design should not be extended, therefore the need for fast and flexible creation and modification methods of 3D objects is apparent.
- Information-handling: The product data model should be capable of handling all required information of the

early ship design process. In addition, the PDM must be able to handle the increasing granularity and population of the model information. Especially the information needed to check the technical and economic feasibility should be stored.

- Data-sharing: The coordinated storage of information assures the designers that all parties involved in highly parallel tasks work on the same data. Additionally it is ensured that the information stored in the PDM can be accessed by the numerous specialized design tools via a well-defined interface including an open object oriented application programming interface (API).

- Requirement-tracking: The compliance of requirements needs to be checked. In addition it should be possible to trace the objects which are responsible for the compliance of a requirement. Therefore input and storage of requirements in computer as well as human readable form needs to be possible.

2.3 MODULAR SYSTEM ARCHITECTURE

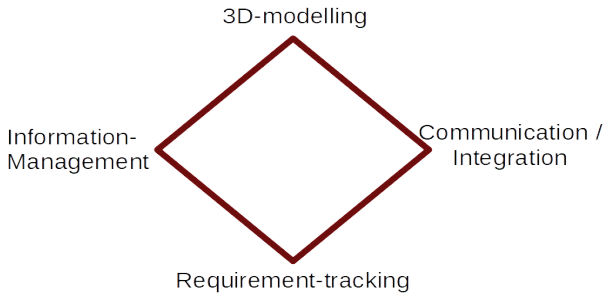


Figure 3: Main functions of the design system

The best system would be optimal for all functions shown in Fig. 3. This kind of system has not been developed yet. There are systems adequate for some functions but containing weaknesses in other aspects. The system architecture presented in this paper is based on a modular approach which allows the usage of tools with the best possible efficiency, whereas the advantages of the individual tools can be combined by connecting them.

The essential requirements in Fig. 3 are “3D-modeling” and “Information-handling”. In concept design an efficient performance in the creation and manipulation of geometry is of importance. In order to check the technical and economical feasibility of the design the product data needs to be stored and made accessible in an efficient manner.

Following an intensive market research it became apparent that there is no software system available which is best suited in these two requirement fields. In order to apply the best suited tools a modular interfaced software architecture is introduced.

In this software architecture two core modules shape the backbone of the system. A two step evalution process was carried out to find the best tools for the core modules. The evaluation and assessment is discussed in detail in [3] and [4].

The results of the evaluation process suggest Bentley AECOsim Building Designer as 3D-CAD-module. The modelling functions of this multi-purpose CAD system are well developed and best serve the geometry creation and manipulation tasks in the concept design. This includes the capabilities to derive and configure drawings from the 3D CAD model which are adopted to the specific needs of shipbuilding.

As module for the information handling a PDM system is interfaced. The PDM system uses a database management system to store design information. A single centralised database server avoids redundancy of data and ensures the availability of up-to-date data. In order to link the geometry in the CAD system with the information in the PDM system and make the information available in 3D CAD system a bidirectional interface between these modules is necessary.

The modules associated to the requirement fields of Fig. 3 are defined as follows:

- *3D modelling: Bentley AECOsim Buildingdesigner*
  This module is used for the geometry modelling. Within the CAD system all geometrical information such as distances, position, area or volume are calculated.

- *Information-Management: SQL Database Server (MySQL, ORACLE)*
  The database itself is any SQL server. The configuration and management of the information is done with an object relation mapping (ORM) system. This system also takes care of the mapping between the relational database and the object oriented API of the CAD system.

- *Communication/Integration*
  This module is part of the common API of the CAD system and the ORM system. As it depends on the shipyard’s, which specialized tools are used and which needs to be interfaces it has be easy to develop interfaces.

- *Requirement-tracking*
  The requirement-tracking is integrated in the PDM-system. The data model is capable to store requirements as objects. Functions are developed to check these requirements against a design. The requirements will be imported or defined via specialized interfaces.

2.4 SYSTEM ARCHITECTURE

This chapter will briefly explain the system architecture, details can be found in [3].

The 3D CAD system is used as storage for the geometrical data and geometrical model operations, whereas the PDM system stores all additional data. Data redundancy is reduced as scarcely geometrical data is stored in the PDM system. For that the real-world object is split into two virtual objects. One object is the geometrical object in the CAD system. The other is an object in the PDM which holds the identity and any non-geometrical information. This PDM objects contains a unique identifier to the geometrical representation. This implies the prime importancy of robust, reliable and fast identification and synchronisation of objects in both directions between the systems. In addition, the PDM system needs to be seamlessly integrated in the CAD system so that the user is on one hand not hindered in his work-flow by an overcomplicated interface and on the other hand can make efficient use of the information. For that reason a object relational mapping (ORM) system was chosen which is integrated in the API of the CAD system. Bentley AECOsim Building Designer offers an API in .NET with the COM-Interop technology.

As .NET is widely used in multiple industries including web-server applications there are a few ORM systems to configure and access SQL server. In this system architecture Telerik Dataaccess was used, because it allows the usage of different types of SQL server and additionally can evaluate and modify schemes of a SQL server.

In Fig. 4 the developed system architecture is shown. The CAD module is located on the left hand side and the module for the information-handling is located in the centre. On the right hand side there are two possible options shown for interfacing specific software tools in shipbuilding.

The two databases (3D-model and the SQL data storage) combined form the 3D general arrangement. The bidirectional interface between these two components is developed in .NET and integrated in the API of the CAD module. Events automatically detect changes in the geometry objects in the 3D model and update the corresponding data in the PDM.

The shipyard specific data model is defined/modified in the ORM software. This software generates on the one hand the class libraries for the .NET integration. On the other hand the necessary files (DDL scripts) for the setup/modification of the SQL server are provided. Because the data model differs between shipyards and shiptypes an ORM software was chosen, trough which the data model can be adapted to the needs of the shipyard.

The system architecture was set up in that way that only the data model needs to be modified by the shipyard. The SQL server and the modifications in the library are applied automatically. Additionally most of the forms for the GUI integration in the CAD system directly extract their information from the library, and thus adapt to the modified data model automatically.

In Fig. 4 on the right hand side the upper possible interface suggests the usage of the class libraries of the ORM system directly in the API of the shipbuilding specific software tool. This is only possible if the API is based on .NET as well. Nevertheless this is the preferred interface as it allows the shipbuilding specific software to use all information available and ensures up-to-date data.

The lower work flow relies on generated or exported files and command scripts which are then processed in the shipbuilding specific software. This is the most versatile way for an interface because almost all software tools have an API based on batch command scripts. Their disadvantage lies in a relatively high effort to develop export routines in the ORM system.

In Fig. 4 no part is designated to the requirement tracking field. The field of system requirements is part of the information handling and therefore located in the middle section of Fig. 4. The modifications to the data model are discussed in the following chapter.

A system infrastructure based on a centralised database server which is integrated in the client CAD tools reduces the data redundancy significantly. This allows every designer involved to use the latest data as design changes are immediately synchronised to the server. This is also valid for the interface for specialised software tools. The derived models in these tools are representing the actual state of the PDM.

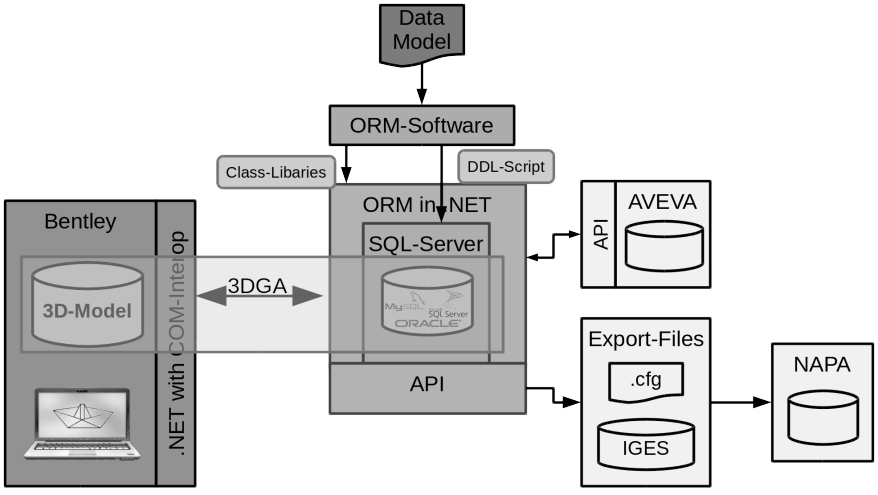


Figure 4: System architecture

3. REQUIREMENT FORMALISATION

The ship concept design is a creative and individual process. A design is developed according to the needs and requirements of the owner and is influenced by the regulations. Any sort of requirement management has always been part of the concept design process but there where no standardised procedures available. The demand for standardised procedures in the requirement management increases also in the commercial shipping industries. This trend is also represented by the process analysis, where a top level field addresses requirement management.

In order to automatically check requirements it is necessary to have a formal description of the requirements implied. Another constraint is that only requirements which test information, which is represented in the design data model can be checked. As the design or rather the data model of the design is based on objects, e.g. components or rooms/tanks, the requirements have to be broken down to this level as well. If a requirement demands for information which is not apparent in the data model, this requirement must either be redrafted to match the data model or the data model has to be adapted to be able to contain that information.

3.1 REQUIREMENT BREAKDOWN

The requirement breakdown is the essential part in the requirement management. This process is described in numerous papers like [5]. The requirement breakdown in the early ship design has been described in detail previously in [6]. This process starts with the owner´s requirements including all derived regulatory requirements and standards. These requirements are written in the voice of the stakeholder (also referred as voice of customer (VoC) in [6]). In the shipbuilding industry this is typically the owner and statutory documents. The shipyard has typically no influence on the wording of the requirements.

The first two phases of the requirement breakdown (requirement elicitation and analysis) are to understand and negotiate the customer´s requirements. The goal of theses steps is to find a common understanding of the requirements between shipyard and owner. The next step is the requirement specification, where the customer requirements are defined and specified by the shipyard in their own voice. In this step the requirements are broken down to functional requirements which should at least have the following quality aspects:

- Unitary
- Complete
- Verifiable
- Unambiguous
- Non-Conjugated

Functional requirements demands for characteristics of certain objects and is in this paper referred to the component level.

3.2 DATA MODEL FOR FORMAL REQUIREMENTS

Requirements exist in different forms and can be of high complexity. Simple requirements guarantee a number of certain well defined objects, such as number of containers, whereas complex requirements can demand the conformity with regulations like SOLAS.

As the requirements to be checked are formulated by the designer the requirement can be assumed unambiguous, testable and solution free. According to [7] a requirement can be interpreted as a demand for an object. This leads to the logical split of the requirement into a descriptive part of the object and a demand to that object as shown in Fig. 5.

In Fig. 5 a simple functional requirement on the component level is shown. The first line shows the requirement text which will be recorded in the documents. In the second line the parts of the sentence are marked which are the description and the demand to the object. Below the description and the demand the necessary properties of them are listed with the values given in the example text.

The description contains at least one object class. The description is refined by adding an arbitrary number of attribute, comparator and value combinations. In this example the tanks, for which the demand is applied, are described in more detail by the purpose type of the tank to be equal to freshwater. This means all objects of the class `Tank` with the attribute `tankpurposetype` equal to `freshwater` are selected for the evaluation of this requirement.

All Tanks with tankpurposetype equals freshwater should have in total a Volume greater than or equals 70m³.

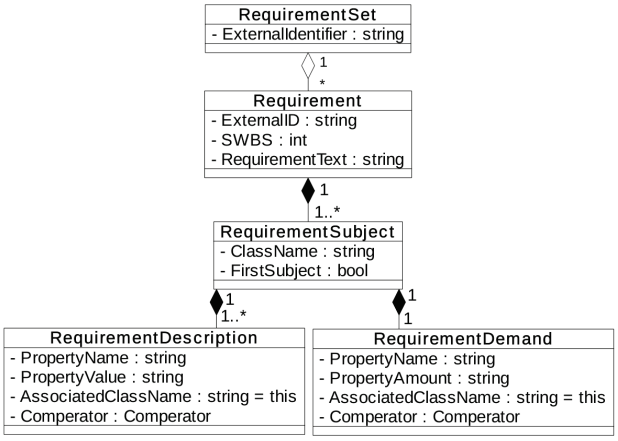
description		demand	
Objectclass	Tank	Demandtype	PropertyAmount
Attribute	tankpurposetype	Attribute	Volume
Comparator	equals	Comparator	greaterThanOrEquals
Value	freshwater	Value	70m³

As shown in Fig. 5 the first characteristic of the demand is the demand type. There are various different types of demands. In the system developed so far two demand types are implemented. The first is the object number, where all objects which are described by the requirement are counted and compared to the demanded value (e.g. number of beds). The second demand type is the need for an amount of a certain property of the objects (e.g. volume of a tank). In this example a need to the property `Volume` of the eligible objects is formulated. If this demand type is used a combination of attribute, comparator and value is used as the demand.

In Fig. 5 a simple example is shown, where the description is defined by properties of the object itself. The complexity increases when the description of the objects is not only defined by properties of the object class but by associated classes. For example an element which is described by the room it is in, like "An element of type Bed in a room with type Crewarea_cabin should be installed 20 times.". This is necessary depending on the data model. The data model for storing requirements should be as generic as possible to allow modifications to the product data model at later stages.

This leads to a data model for the requirements as shown in Fig. 6. The class diagram starts at the top with a class called `RequirementSet`. This class groups together some requirements. This class is also used to store an identifier to external software, if the requirement management is organised in special software tools like Rational Doors [8].

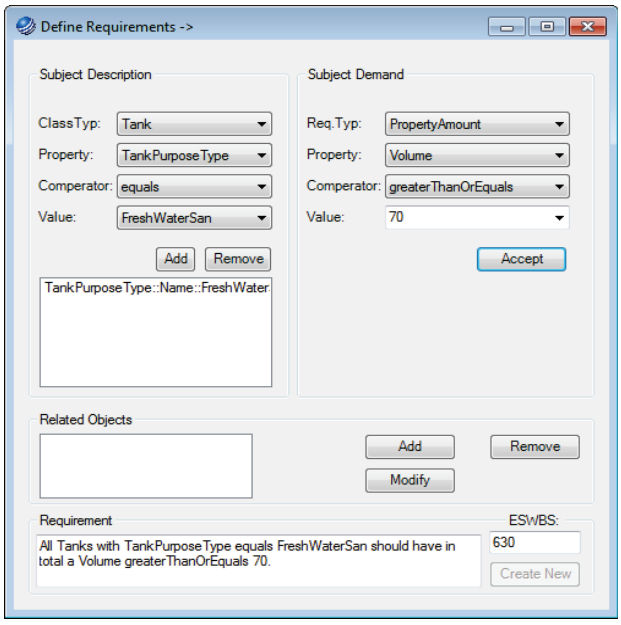
The `Requirement`-class in Fig. 6 is representing the actual requirement. This class has an "ExternalID"-property to store an identifier to a requirement in external management software tools. Additionally there is one property to store the requirement text and another one to store the identification in some sort of organisational system identification scheme. In this example a Ship Work Breakdown Structure (SWBS) was used. The `Requirement`-class is a composition of one or more `RequirementSubject`-class entities. One `RequirementSubject`-class entity is representing a single statement like the example requirement given. A requirement can consist of multiple connected statements. Nevertheless the first described object will remain the main subject of the requirement. Most of the requirements have just one `RequirementSubject`. The `RequirementSubject`-class consists of any number of `RequirementDescriptions` and one `RequirementDemand`.



The `RequirementDescription` and `RequirementDemand`-class has the above described properties to store the combination of property, value and comperator as described above.

3.3 REQUIREMENT DEFINITION BASED ON PRODUCT MODEL

As described above a formal way of implementing requirements is shown. In order to be able to automatically check these requirements against the product data model it is required that the class and property names are correctly set in the requirement. This leads to a sensitivity against misspelling due to the "string"-data type of the `Classname` and `Propertyname`-attributes in Fig. 6. To avoid mistakes in spelling by the user and to facilitate the creation of requirements a form was developed to assemble the requirements from prearranged lists. This form is shown in Fig. 7. On the upper left side of this form the description is formulated by multiple `Property`, `Comperator` and `Value` combinations. These are added and removed from the list via the buttons. The demand is set on the upper right hand side. In the middle section ("Related Objects") additional `RequirementSubjects` can be added. In the lower part of the form the requirement text is built together from the data given above. This text can be modified in case the



automatic wording is not sufficient. Nonetheless modifications to the text are not modifying the representation of the requirement in the data model.
The form shown in Fig. 7 is mainly based on drop down lists to choose from. On the one hand this speeds up the requirement input process. On the other hand this avoids misspelling of class and property names. The drop down lists are extracted from the product model on every start of the form.

The product data model is changing over time and new properties and types of components are added regularly. As the data model is analysed on every start, the form is adapted to these changes without any modifications. This means that there is no doubled work when introducing new classes or properties, as well as no redundancy of information which is prone to mismatch.

In the first step all classes of the product data model are filtered. According to Fig. 4 any information of interest is inserted trough an interface into the PDM system. The algorithm excludes all non-public classes as they are not of interest. Afterwards classes are excluded which are not subject to requirements, like comments, project management or for the requirement management itself.

In the second step the properties of the filtered classes are analysed in a similar manner. In a first loop the algorithm includes only simple type of attributes, like `double`, `integers`, `strings` and `booleans`. In a second loop it excludes attributes via list to get a short and easy searchable list in the drop down list. In a third loop it adds the properties of associated classes if needed.

In Fig. 8 an excerpt of an exemplary product model is shown. The procedure explained above will find the `Element`-class. This class will be added to the drop down list of classes. Afterwards the drop down list of properties will be filled with the `Element`-properties. Since `AssemblyIdentification` is on the list for including the properties in the `Element`-class it will not be added to the class drop down list. However the properties of the `AssemblyIdentification`-class will be added to the properties drop down list of the `Element`-class. Properties including the substring "`_id`" and "`BentleyID`" are omitted.

The resulting items of the property drop down list of the `Element`-class are:

- `ElementType`
- `Name`
- `Shortname`
- `Annotation`
- `IsDeadweight`
- `COVX`
- `COVY`
- `COVZ`

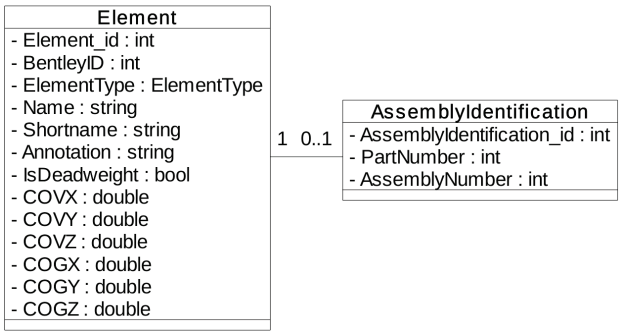


Figure 8: Excerpt of class diagram of Element and AssemblyIdentification

- `COGX`
- `COGY`
- `COGZ`
- `PartNumber`
- `AssemblyNumber`

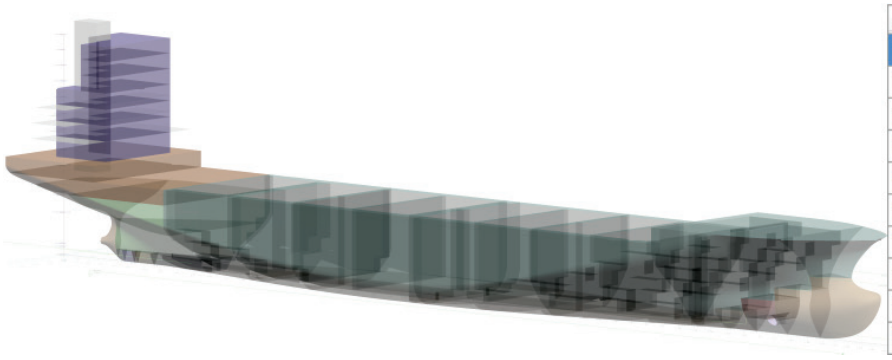
This information is used to check the requirement against the product data model. The algorithm to check requirements reverses the described procedure. It searches for classes in the product model which have a name equals the class name given. Afterwards every object of this class will be queried. These objects are then filtered by the property, which is defined by the property name. The value of the property is compared to the given value in the description of the requirement.

To check if the requirement meets the demanded value the sum of the described property in the demand is calculated over all found objects. If this accumulated value is in the correct relationship, given by the comparator, to the demand value this requirement will be marked as "comply".

4. CASE STUDY

This section provides case studies of how design tasks are performed based on the system architecture. The tasks are related to Fig. 2. The integration in the GUI of the CAD system is exemplified in Fig. 6. Each object has an update-method implemented which will be executed when the event-handler of the CAD system notices changes to the object.

The first case study will demonstrate the flexibility in creating the compartmentation. This is mainly based on the modelling functionalities which are part of the CAD system. Additionally, the automatic update going along with changing the geometry will be shown. The second case study shows the task "placing components" which is an important step in the concept design. In the third case study an example is given where a requirement is formulated and checked.



Item_id	Name	Volume
123512	Leakwater Tank	4,174
123661	Main Engine Room	1127,506
123690	Main Engine Room	3446,496
123808	ContainerHold; 116 Container	4640
123819	ContainerHold; 120 Container	4800
127393	Fore Peak Anchor Store	626,236
127454	ForePeak Ballastwater	439,369
128690	Ballast Water	342,522
128717	Bow Thruster Room	187,137
128730	Seachest	56,057

Figure 9: Compartmentation of a 2500TEU container ship design. Engine room and deck house not further subdivided.

4.1 CREATE COMPARTMENTATION

One of the first major steps in design is the subdivision of the whole ship body. This describes the first idea of the different functional areas of the ship. This task is a time-consuming step in the advanced iterations when it comes to e.g. designing complex shaped rooms or adding smaller rooms to the centre of the ship. In Fig. 9 the compartmentation of a small 2500 TEU container ship is shown. This compartmentation was done in about an hour and consists of 85 rooms/tanks. The decks and bulkheads were generated by slicing the ship's volume. The more complicated container holds were modelled outside of the ship, then placed in the ships volume and afterwards subtracted from the ships volume. This procedure speeded up the generation of recesses like the shown monkey seats of the container holds. On the right side of Fig. 9 a list with an excerpt of the information about the rooms/tanks is given. This list represents some data from the PDM and is integrated in the GUI of the CAD System.

4.2 PLACING COMPONENT

Fig. 10 shows the design of a fishery protection vessel. The model is populated with equipment as well as accommodation and sanitary furniture. Each equipment has attributes such as name, weight, type and centre of volume. The form needed to assign these attributes appears on the right side of Fig. 10, where the attributes of the port rudder are shown. The system automatically detects in which compartment the equipment is. As this information is assigned to the equipment, it is easily possible to list equipment in a room or tank.

4.3 AUTOMATED REQUIREMENT CHECKING

In Fig. 7 the methodology to input requirements is shown. This form is either implemented in the specialized requirement management tool or can be used as stand alone. The formulated requirements are then imported into the 3D general arrangement tool to give the designer an overview about the requirements he/she needs to work on. If required the designer is able to trigger the check for a single requirement or the requirements altogether. Some visualisation features have been implemented to easily trace back the objects that fulfill the requirements.

An example is shown in Fig. 11. On the right side the form for managing the requirements is shown. In this list four requirements have been saved. The second requirement of the list is the example given above. In the actual design there is enough technical fresh water tank capacity, but the assigned tank for sanitary fresh water does not have the required volume. The location of these tanks in the ship are shown on the left side of Fig. 11 in blue colours.

5. CONCLUSIONS

In this paper a ship design system based on a 3D-model for the concept design is presented. Due to the modular system approach a 3D CAD system is used to manage the

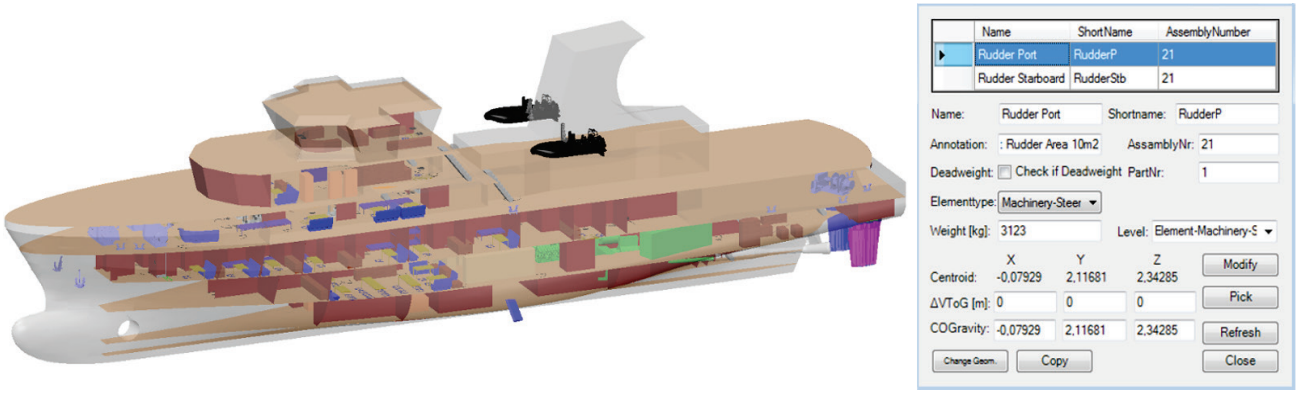


Figure 10: Example for placing equipments. Fishery protection vessel populated with equipments.

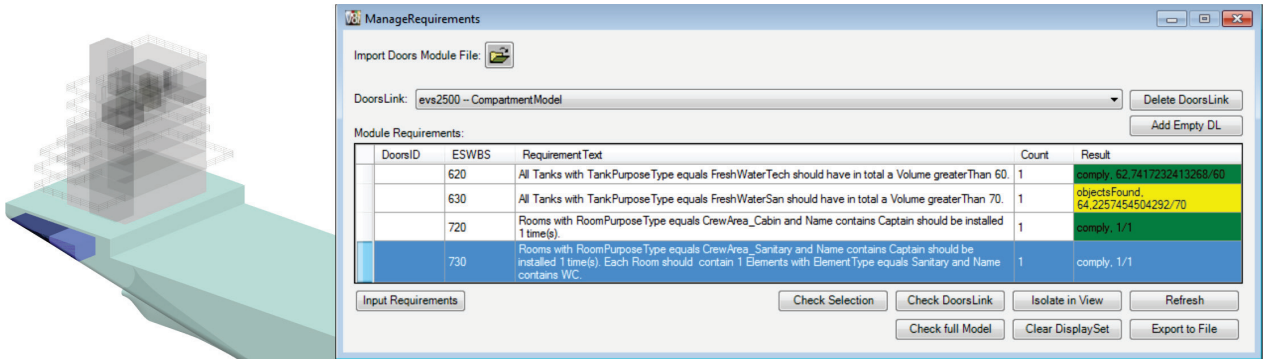


Figure 11: Example of the requirement handling.

geometry. This allows the generation of the 3D general arrangement in the very early design stages without violating the typical time horizon. Accurate geometrical information is calculated and saved in the database. Together with additional data the product knowledge is increased which leads to the reduction of risks in the design.

The presented modular system architecture with the interface options for specialized tools supports the design ultimately. The integration of this software tools avoids out-of-date data and reduces the risks of errors during manual data-transfer and also accelerate the model creation in other tools.

The examination of the design is simplified due to a higher amount of information available in the PDM system. The methodology to check the design with respect to requirements and trace back the associated objects described in this paper builds up confidence in the design. This increases the efficiency in the concept design and therefore excites new developments.

6. ACKNOWLEDGEMENTS

We would like to thank the Flensburger Schiffbau-Gesellschaft and ThyssenKrupp Marine Systems for their valuable support. This work was funded by the German Federal Ministry of Economic Affairs and Energy (BMWi) based on a decision of the German Bundestag, grand No. 03SX336B.

7. REFERENCES

[1] M. Zimmermann, ‘Knowledge-Based Design Patterns for Detailed Ship Structural Design’, University of Rostock, Rostock, 2010.

[2] D. J. Eyres and G. J. Bruce, *Ship Construction*. Butterworth-Heinemann, 2012.

[3] H. Lindner, S. Schenk, R. Bronsart, B. Ebeling, K. Frömming, and F. Kluwe, ‘A Modular System Architecture for the Early Ship Design by Combining a 3D CAD System with a Product Data Management System’, in *14th International Conference on Computer and IT Applications in the Maritime Industries*, 2015, pp. 135–146.

[4] R. Bronsart, H. Lindner, and S. Schenk, ‘Entwicklung einer 3D-topologischen Produktkoordinierung zur Optimierung des Generalplan-Entwurfs komplexer Schiffstypen - Abschlussbericht’. 2015. (in german)

[5] J. (Roger) Jiao and C.-H. Chen, ‘Customer Requirement Management in Product Development: A Review of Research Issues’, *Concurr. Eng.*, vol. 14, pp. 173–185, 2006.

[6] S. Schenk, H. Lindner, R. Bronsart, K. Frömming, and B. Ebeling, ‘A contribution to the Requirement Formalisation in the Early Stage of Ship Design’, in *12th International Marine Design Conference 2015 Proceedings Colume 3*, Tokyo, 2015, vol. 3, pp. 146–157.

[7] L. v. Ruijven and U. Nienhuis, ‘Requirement management, traditional and a second generation scenario’, in *PROCEEDINGS OF COMPIT ‘06*, Oegstgeest, The Netherlands, 2006, pp. 286–301.

[8] Ibm.com. (2017). *IBM - Rational DOORS*. [online] Available at: https://www.ibm.com/software/products/en/ratidoor [Accessed 24 Aug. 2017].

8. AUTHORS BIOGRAPHY

Robert Bronsart is holding the Chair of Naval Architecture at the Department of Mechanical Engineering and Marine Technology at the University of Rostock. He has more than 35 years experience in R&D in ship design, production and operation, thereof ten years in industry. In research he is focusing on information and communication systems in collaboration networks, CAE-system integration and on knowledge based methods in ship design and operation.

Hannes Lindner is a research assistant at the Chair of Naval Architecture at the University of Rostock. Prior to that he worked on research projects focused on the early ship design process. He graduated from the University of Rostock with a Diploma in Mechanical Engineering, specialised in Naval Architecture and Ocean Engineering.