Cloudless Skies in Early Design?

Thomas Koch, Atlantec Enterprise Solutions GmbH, Hamburg/Germany, thomas.koch@atlantec-es.com
Konstantin Kreutzer, Atlantec Enterprise Solutions GmbH, Hamburg/Germany, konstantin.kreutzer@atlantec-es.com
Andreas Roppelt, Atlantec Enterprise Solutions GmbH, Hamburg/Germany, andreas.roppelt@atlantec-es.com

Abstract

One goal of the SHIPLYS project is to increase efficiency and integrity of early design processes while maintaining full flexibility, creating additional space for creativity and improving affordability, in particular for SMEs. We present the approach chosen to realise a fully distributed design environment of loosely coupled design tools utilising REST-based services, reviewing the architecture, services, interaction models and integration principles that have been conceived to enable designers to execute complex design processes effectively and yet more thoroughly. We also look at security and data protection facets, which require careful consideration in distributed systems.

1. Introduction and Background

The European shipbuilding sector, with more than 300 shipyards and a network of over 9,000 subcontractors (mainly SMEs), accounts for a total of about 350,000 jobs and a turnover of around €34 billion per year, *Bharadwadj (2017)*. Nevertheless, this sector is facing a global competition steadily becoming fiercer with the strategic investments in ship building and repair being made in the Far East and low labour cost countries. Furthermore, client expectations enforced by additional rules and regulations coming into force, for vessel quality, pollution controls and through life performance are becoming more sophisticated. Consequently, in order to provide the best offer during the bid stage, it is indispensable to be able to create innovative designs and at the same time, reliably predict the innovation's impact on building costs, ship quality and operational performance.
Major progress has been made in the fields of numerical simulations, computing technologies and efficient organization of distributed working environments. Yet, the calculation and modelling to do in order to take advantage of this progresses is difficult and time consuming – especially for SMEs without a large overhead of trained staff and tools – due to difficulties in integrating data flows between incompatible tools and formats for different design stages. To a certain extent, SMEs are limited in their ability to make use of technological advances while innovations are considered as a critical precondition for survival. In order to apply innovation with economically acceptable risk, there is a need to **reduce the cycle time and costs of designing processes** (especially during bid preparation stages). This will enable an SME to

- develop multiple alternative designs, leading to an improved final design,
- perform more thorough risk assessment at earlier stages of design,
- increase the level of confidence in the innovation-based ship's performance and costs across the complete product life-cycle (from design to production, operation, … up to the decommissioning)

This is achievable by decreasing the amount of time and financial resources needed for the early design which should facilitate more detailed pre-bid-designs which can be assessed more thoroughly (e.g. by performing early numerical studies, production simulation, LCA etc.) .

SHIPLYS has proposed to address these issues by applying three main concepts, *Bharadwadj (2017)*:

- **introducing a framework of distributed services (allowing exploitation of multi-level synergies) to establish a state-of-the-art design tool chain:**
  This concept aims to solve several problems such as providing cost-efficient access to sophisticated tools and methodologies, i.e. enabling SMEs to use otherwise unaffordable technologies on a job-based costing model, simplify design iterations and re-execution of calculation jobs creating updated versions of affected data. Also, a project-wide infrastructure for shared access of monitored and versioned data increases collaboration effectivity and the level of overview, *SHIPLYSD31 (2017).*
- **allowing more effective reuse of previously generated information (reducing cycle times dramatically):**
  This is inspired by other industries efforts and experiences in identifying and capturing the useful implicit information already present in existing CAD/CAE data. In particular, the Building Information Model approach, *BIM (2011)* increasingly adopted in the oil & gas and construction sectors appears to be based on promising principles such as product- and life-cycle-oriented data standardization.
- **introducing 3D-models in early design by enabling automatic data generation/rapid virtual prototyping (allowing to investigate alternative designs):**
  The last concept takes advantage of the previous two aspects and aims to drive pre-bid design much further, including preliminary production simulations based on 3D-prototypes.

In order to validate these concepts, three scenario types have been chosen to be investigated. Scenario 1 addresses the comparison of innovative alternatives as well as the evaluation of a design's life-cycle impact as it is about optimisation of designs for a short-route ferry using hybrid propulsion. In scenario 2, the full potential of SHIPLYS framework will be applied on the newbuilding design of a multi-purpose carrier. Scenario 3 should show the applicability of the SHIPLYS framework to retrofitting and repair cases and deals with the retrofitting of a scrubber system to a ferry.

Looking at the underlying information technologies, hardware and software developments continue at a rapid pace, while the cost of their use has stabilised at low levels. Nevertheless, as complexity and variety of systems and solutions increases, it cannot be considered an easy task at least for smaller organisations to deal with these advances in an efficient way. An example is the introduction of "cloud" technology, where marketing efforts of vendors might have created more confusion than assurance, not at least due to the increasing concerns about cyber-security. For organisations that are dependent on protecting their intellectual property to maintain their competitive advantage and specialised know-how, the common perception of "The cloud" creates many unanswered questions – if not threats – and thus inhibits the unconcerned adoption. What is commonly overlooked in this discussion is the fact that these propositions are based on several powerful technological developments (often summarised as cloud technologies), for example virtualisation, *Virtual (2006)*, containerisation, *Docker (2017)*, provisioning and scalability, *OpenStack (2017)*. These technologies provide a great potential for improvements without the actual ultimate need to utilize remotely operated 3rd party cloud services. However, to accomplish this, the use of these technologies needs to be tailored to the needs of the business users.

2. Improving Early Ship Design Processes

The SHIPLYS design solution's ambition is to maintain and improve the competitiveness of Europe's SMEs in the maritime industries, especially small- and medium-sized shipyards and engineering offices by enabling them to generate offers that are thoroughly investigated. To allow investigation of various aspects of a ship's performance over the entire life-cycle or to assess inherent production risks (e.g. time schedule, capacity, and technical demands) as well as a dependable estimation of design and production costs, a simulation-based approach has been chosen. Obviously, such kinds of simulations have to be implemented using three-dimensional models. But how do we get there before the bid?
One important component of the SHIPLYS approach is to formalize, capture/store and reuse the explicit and implicit assumptions and decisions at conceptual design stage that are part of the design

engineer's vision and nowadays often not documented. This already begins even before conceptual design where we introduce tools to identify the requirements of the ship based on specifications or tender documents, store them and evaluate the degree of the design's compliance automatically. Formalizing design decisions can only be done when using a comprehensive, industry-specific data model that is capable of capturing all relevant facets of the product. For this purpose, we are using a data-model that takes advantage of the effort invested in various standardization procedures. The vision of this concept is a project environment where integrated tools can access and process every relevant piece of information generated during the project.

Another idea we are pursuing is to formalize the design process itself by means of an attributed process model, *Koch (2017)*. It is based on a graph structure of design activities related by their required or optional inputs and generated outputs, *Brandes (2002)*. Such a model provides the basis to perform repeated dependency analysis operations to identify design activities that are ready to perform or to identify the missing data that is needed to perform a specific design activity. This way, we can monitor the progress of the design process and provide suggestions for pending operations. Furthermore, it facilitates the identification of dependencies resulting from outdated calculation results, e.g. due to design changes. It also enables us to identify lack of data or missing tool support, where target-oriented automation would allow driving the design process much further. With this preparatory work done, we aim to develop rapid virtual prototyping tools or use functionality of already existing software to close the identified gaps, often by generating three-dimensional models. Example use cases could be:

- generating a (preliminary) three-dimensional hull form from given main parameters,
- generating (preliminary) mid-ship section and main structural elements from hull form and ship purpose,
- generating (preliminary) block sections and assembly structures from structural elements,
- … and many more applications.

As this approach allows including sophisticated design tasks such as operation-related calculations and production simulation, another important target of SHIPLYS is to perform full life cycle impact analyses. To simplify and improve the evaluation of a design's overall quality compared to another design, a multi criterion decision making support technic should be developed as well. Last but not least, the concept of loosely coupled services and clients allows efficient distributed working on the project among many participants.

3. Integration and Information Management

3.1 Integration of Tools

Since the early design is commonly dependent on a diversity of standalone software components (often with limited data exchange capabilities) an important element of our approach is the creation of a framework for integration of various software components. These components may cover a wide range such as full-scale applications, analytical programs, smaller or larger utilities, database management systems or other data sources, and software libraries. Such components originate from 3rd parties or end-users (e.g. utility type tools developed for the specific needs of a design scenario).

In general terms, integration of such components must be based on the following assumptions:

- In many cases modification of an existing component is either not possible or impractical. This is particularly true for commercial 3rd party components which may often appear in the form of a black-box type of software with no access to sources and limited configuration options. Such components will often mandate a specific runtime environment, which cannot be changed. As a consequence, integration should be possible without any real modification of such components.
- Components will often operate on input and output data streams of known formats combined with interactive control by the user, as is the case, for example, for applications performing computationally intense calculations. It must be assumed that sufficient documentation or

similar explanatory material will be available such that some standardised "glue" code can be provided in the project for such components to provide the required input and control data or consume the output of interest.

- Alternatively or additionally, software components or tools may have their own data management capabilities. For this kind of components data access interfaces will often be available.
- A further variant may be constituted by some degree of built-in extensibility e.g. by means of scripting or plug-ins. This is particularly relevant in highly interactive components such as modelling systems.
- It should also be possible to develop new components with minimal constraints concerning the implementation technology, e.g. free choice of runtime platform and programming environment and other technical constraints. This is mandated also by the fact that such components will often be based or be dependent on existing algorithms or sub-components and/or reliance on existing know-how among development staff involved.

Based on these assumptions, the following general design goals have been established for the project, *SHIPLYSD33 (2017)*:

- Scalability (computational capacity, parallelisation, as well as storage capacity)
- Minimised programming technology constraints for software components:
    - runtime platform
    - programming platform
    - language and time zone
- Support for loosely-coupled, distributed operations: components may reside on hosting systems as long as they are reachable via network
- Light-weight integration of black-box type components
- Maximised support of functional requirements of early design processes

For purposes of prototype implementation, the following "Integration Methods" have been defined:

Glue code

This integration method provides functionality to expose component functionality of black-box type software via a SHIPLYS framework client. This Integration Method is particularly useful for non-interactive computation modules, see Fig. 1.

Examples: standalone modules like weight or CoG calculations, resistance prediction, FEM or CFD processors.

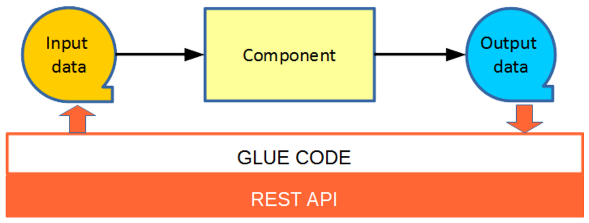

Fig 1: Glue code integration method

Data access interface

This method is applicable to systems operating on some sort of data management platform, see Fig. 2. It can range from simple file system level retrieval to full-scale data management system access.

Examples: CAD or PDM system data retrieval, import, export or drawing extraction.

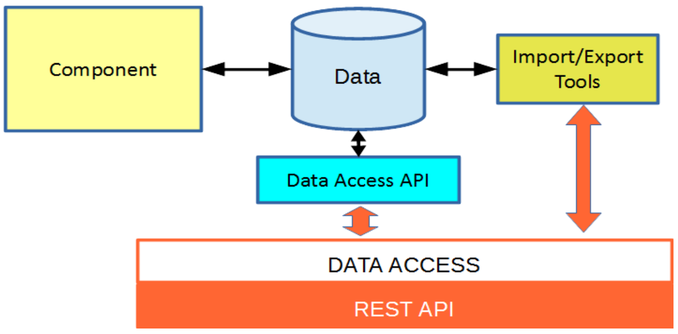

Fig. 2: Data access integration method

Plug-ins
For systems or components that provide mechanisms to extend or tailor their functionality via e.g. a plug-in API or a scripting environment, this method (Fig. 3) can be used to enable such systems to interact themselves with the SHIPLYS framework. For example, this may be realised as menu/user interface additions in user interfaces that effect some data retrieval or trigger a calculation in another remote component.
Examples: many CAD systems, geometry design tools.

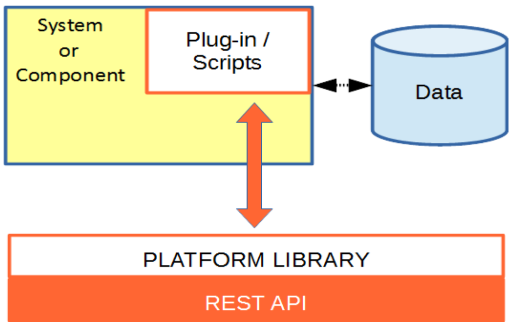

Fig. 3: Plugin integration method

3.2 Features of the Approach

The following main features define important characteristics of the approach:

REST services – the framework is based on a set of services accessible via a REST HTTP API utilising JSON encoded payloads. REST over HTTP relies on conventions layered on top of the ubiquitous HTTP protocol. Strong reasons for using this approach are the scalability of connection services and systems as well as the flexibility in implementation both in terms of actual design of the API itself as well as the relative easiness of support across a wide range of implementation platforms and operating systems.

Process model support – the goal of providing guidance and process management capabilities during the design process is accomplished by providing a process definition and process monitoring capabilities.

Data model support – to accomplish the required self-documenting data model services and to facilitate self-adjusting software components, a meta-data interface as part of the REST API is provided. This interface exposes (sub-)schemes, entities, attributes and build-in low-level type definitions to define the data models in use by a framework implementation. By providing such interface functionality, consuming software will be able to establish references to its local data models.

Data state – to provide data life-cycle support functions including dependency analysis, it is possible to accompany any data entity instance with data state information: version (e.g. as a time stamp), origin (identifying the creating component), quality (e.g. vacant [needed but missing], required, estimated, calculated, validated, etc.)

Function registration and activation – the framework provides lookup and locator functions for software components and the functionality supported by them. Components can register themselves in terms of component identification and description as well as individual operating instances (e.g. CAD system X running on host Y, offering a set of functions Z).

Security – a design project usually involves multiple parties assuming different roles during the project execution. Since commercially sensitive information is created, protection of such intellectual property is mandatory.

4. Process Model

The ISO 10303 standard was started in 1984 as an effort to develop a series of application oriented data exchange standards and tooling methods for the computer-interpretable representation of product information and for the exchange of product data, *ISO (1994)*. While a full scale implementation has been accomplished in several industries, e.g. the automotive and aircraft industry, other parts of the standard have provided the basis for more complex approaches like the Building Information Model (BIM) for the building and construction industry, *ISO (2013*). The adoption in the maritime industry has been somewhat fragmented and inconsistent due to many reasons. Nevertheless, the core content – the detailed data model repository – has served well in various technologies and influences various standardisation efforts such as geometry exchange, *ISO (2001), ISO (2011),* technical document management *S1000D (2016),* or OpenHCM, *HCM (2018).* The shipbuilding related parts of the standard (*ISO (2004), ISO (2003), ISO (2005), ISO (2004a)*) provide a wealth of concise data definitions which have demonstrated their production-readiness after many years in productive operation, e.g. AES (2018).
An interesting, but often overlooked feature of this series of standards is the set of mandatory Application Activity Models (AAM). Thanks to the elaborated synchronisation between the shipbuilding related Application Protocols (APs) by means of the so-called “Ship Common Model”, all activity models in the “Ship” series of standards link together and can be seen as focussed subsets of a complete life-cycle activity model for marine vessels and structures.
All shipbuilding-related APs apply the IDEF0 modelling method, *AFWAL (1981),* to describe the activity functional model. In these models, activities are organised to create a deeply structured hierarchy. The ISO 10303 Application Activity Model for ships (“Ship AAM”) provides a detailed and comprehensive description of the top-most process (A0 – “Perform ship lifecycle”) and its main sub-activities:

A1 – Specify ship
A2 – Complete and approve ship design
A3 – Produce and inspect a ship
A4 – Operate and maintain a ship
A5 – Decommission and disassemble

The early design phase is represented in activity A12 – “Prepare bid” which is structured into some 60+ activities. Based on this initial set of definitions, a design process has been derived and extended, formalised and converted into a fully attributed process definition, *Koch (2017).*

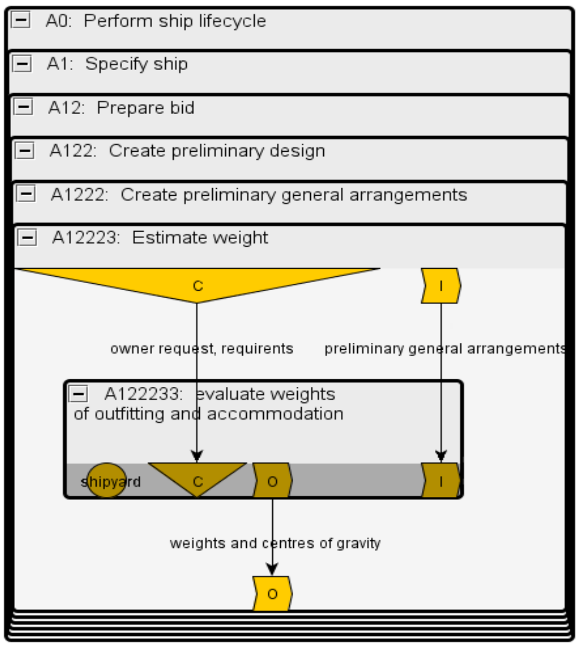
A0: Perform ship lifecycle
A1: Specify ship
A12: Prepare bid
A122: Create preliminary design
A1222: Create preliminary general arrangements
A12223: Estimate weight
C
I
owner request, requirents
preliminary general arrangements
A122233: evaluate weights of outfitting and accommodation
shipyard
C
O
I
weights and centres of gravity
O

Fig. 4: A sample activity definition from the attributed process model

5. Data model

An important element of integration is mutual agreement on the foundation data entities ("business objects") to be communicated between participating software components.

Working with a common data model appears as an attractive and elegant solution. However, based on experiences from precursory projects some consideration must be given to the following issues:

- Beware of the reinvention of the wheel: there are many existing data standards and de-facto standards out there. Many engineering and commercial aspects have been covered, but probably not all and not always at the required level of granularity (either too detailed or too coarse).

- But is a monolithic approach a realistic solution? It would seem an overwhelming task to derive a fully integrated model from this. For this reason the project consortium has taken a close look at e.g. the BIM approach, *BIM (2011)*, which among other aspects is an example of a business domain oriented amalgamation of various contributing standards and rules.

- Project development dynamics are another critical success factor: as several work packages and tasks are advancing concurrently within the project, an intense level of modifications and corrections has to be expected also on the data model side, thus dependencies on progress of other tasks must be mitigated.

A conclusion derived from these requirements and considerations is the need for self-explanatory, machine readable data model definitions that may evolve over time. It means that an implementation should not only provide the classical information management functions for storing, retrieving or locating instances of data entities, but should also provide meta data about the underlying model definitions. This is a well-known concept found in various advanced programming environments as well as in some data management systems with different levels of functionality. In practice the effort to utilise this information tends to be quite high due to the proprietary nature of such systems. Fig. 5 shows how the different layers of data definition and data management are linked.

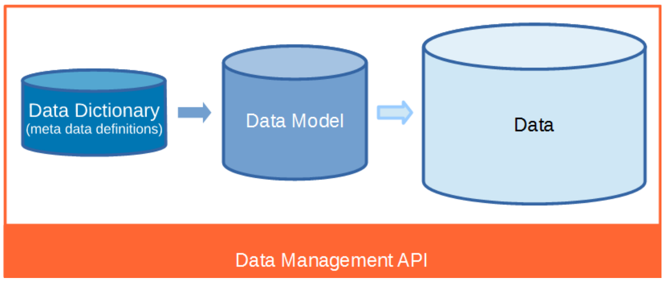

Fig. 5: Data management layers

6. SHIPLYS Framework

Due to the diversity of tools involved in the design process, it is obvious that the shipyard needs described above cannot be met by a monolithic piece of software. Instead an open framework of loosely coupled components is provided to ensure:

- registration of different software tools
- well-defined and secure communication between tools
- concurrent working on the same project
- process monitoring

The SHIPLYS framework shown in Fig. 6 fulfils all these requirements using internet for the communication. It comprises two general types of communicating parties: **clients** representing software components which are requesting information from or triggering execution of tasks by services and **services** representing software components which are servicing requests of clients to deliver information or to execute a task accordingly.

The service types defined for the SHIPLYS framework are:

- **Meta Service**
  This service operates as a kind of dictionary for both data and process models. It provides data models applicable to projects and organizational information managed by data services. It also provides the process model definitions that can be utilized to perform process monitoring, dependency analysis and progress assessment.

- **Data Service**
  These services provide the ability to store and load data for projects, represented in compliance with the data model(s) provided by the Meta Services. Along with the storage functions, data state information is also maintained.

- **Software Registry**
  The Software Registry holds the references to all software tools which can be registered to define the tool chain. Clients interact with the Software Registry to review the list of known software components, to select a suitable tool for a task at hand and to find Job Services (see below) that can execute specific software tools.

- **Job Service**
  Within the framework, many job service end points can be active. A Job Service is a service that waits for an authorized request from a client to perform one or more activities (defined in the process model) using a specific instance of a registered software tool. Such a request will result in the Job Service starting a client that connects the software instance with the framework, requesting necessary data, performing the activity and storing results (by interacting with Data Service(s) to store the results).

- **Authorization Service**
  The Authorization Service provides the fundamental security functions. It manages the association of rights granted to identities and operating permissions applicable to service end points and processes requests for registration of services and authorization requests from clients.

- **Certificate Agency (CA)**
  This is an optional feature: since the framework is intended to be operating over the network, a secure authentication mechanism must be established. Generally, a Certificate Agency provides digital certificates that facilitate the use of asymmetric encryption between end points (clients and services) and provide unique identification /authentication of any certified end point.
  While it is perfectly possible to manage such certificates by utilizing commercial services for this purpose, the provision of a local CA along with the framework components helps to avoid bureaucratic overhead and cost and simplifies operation.
  The fundamental function of the CA service is to take certification requests and to return certificates. It can also be used to perform Certificate revocation in case a previously issued certificate needs to be invalidated. As part of its operation it will also maintain a registry of all certificates issued.

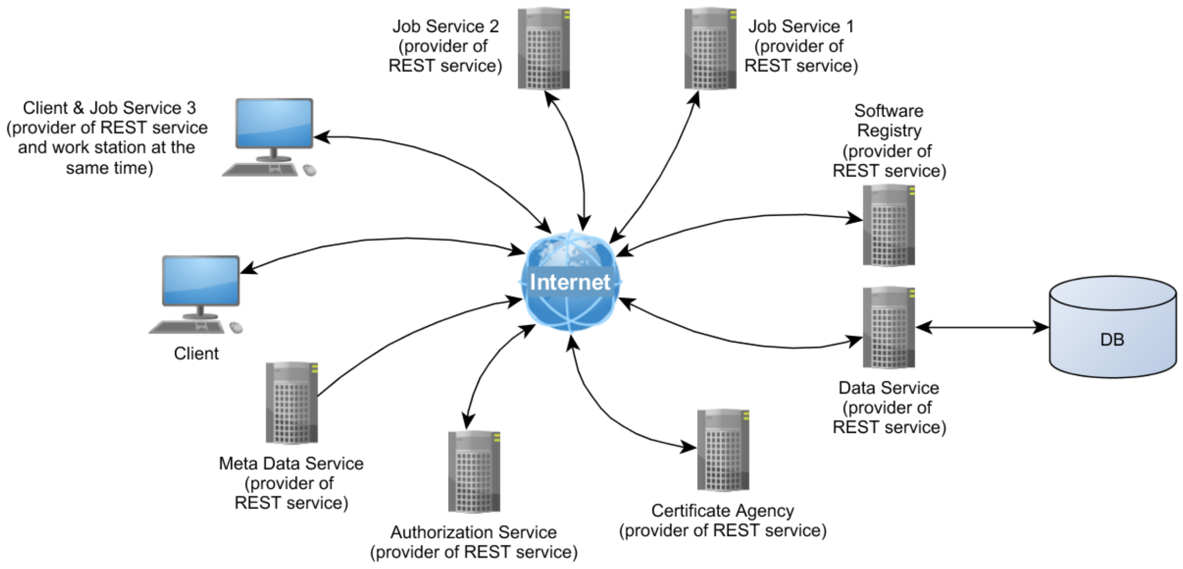
Job Service 2 (provider of REST service)
Job Service 1 (provider of REST service)
Client & Job Service 3 (provider of REST service and work station at the same time)
Software Registry (provider of REST service)
Internet
Client
DB
Data Service (provider of REST service)
Meta Data Service (provider of REST service)
Authorization Service (provider of REST service)
Certificate Agency (provider of REST service)

Fig. 6: Services of the SHIPLYS framework

The core technology applied in the implementation of the SHIPLYS framework is defined as the Representational State Transfer (REST) over HTTP, *Fielding (2000)*, which provides a high degree of platform independence, since only network communication standards apply.

JavaScript Object Notation (JSON), *JSON (2017)*, has been adopted as the payload data format for representing the request and response content. This is not mandatory and can be complemented by other types of payload as required.

Security features are essential in order to ensure secure network-based data transfer to prevent unauthorized access to the project data and use of services. The two main functions provided by the security related components are authentication and authorization which have been realized within the SHIPLYS Framework.

- **Authentication** provides support to answer to the question "*Who is the user?",* i.e. to certify the identity of an acting entity (e.g. a user running a client or a service acting autonomously or on behalf of a user). In order to protect confidential information that is passed over the wire, a mechanism is required identifying the acting entity and supporting message signatures. Such a mechanism is provided by the Transport Layer Security protocol TLS, *RFC (2008)*, which – when combined with the Hypertext Transfer Protocol HTTP, *RFC (2014)* – results in Hypertext Transfer Protocol Secure HTTPS. In order to provide the highest possible security level and to prevent that random clients get access to the framework end-points, the mutual authentication is used after a client has been successfully registered.

- **Authorization** provides the functionality to answer the question "*Is a certain identity authorized to perform a certain operation on certain protected resource belonging to a specific service?*" For networked services, the dominant security method today is OAuth 2.0 *RFC (2012)*, an industry-standard protocol for authorization and therefore has been used within the SHIPLYS framework. Depending on the specific case the OAuth 2.0 describes several main approaches, called grant types, how to protect the resource server. For the framework the grant type "`Assertion`" has been selected as the most suitable and well combinable with the chosen authentication mechanism as both of these mechanisms use certified documents. Using the `Assertion` grant type a client provides a certificate issued by the certificate agency to the authorization service. After a successful verification of the certificate the authorization service generates an appropriate access token containing access rights and provides it back to the client. Finally, the client provides the access token to the protected resource and after a successful token verification it gets access to the API.

7. Prototyping

For prototyping purposes, a demo environment has been established as a proof-of-concept, Fig. 7. This is based on a demonstration scenario that starts with a tender document for a multi-purpose carrier. The design process is initiated by using a tool for requirements identification and evaluation. After formalizing requirements, the design is continued by defining further basic parameters a conceptual design tool, which generates as one result the hull definition. This hull form is processed by RSET, *RSET (2013)*, a compartmentation definition and optimization tool. Hull form and compartments are then used to define the main frame section and other structural elements, to estimate weights and to perform basic strength calculations in the ship design software CAFÉ, *CAFÉ (2011)*. All of this information is finally used in SEASAFE to calculate hydrostatics, *SEASAFE (2017)*. The complete process can easily be performed in one day.

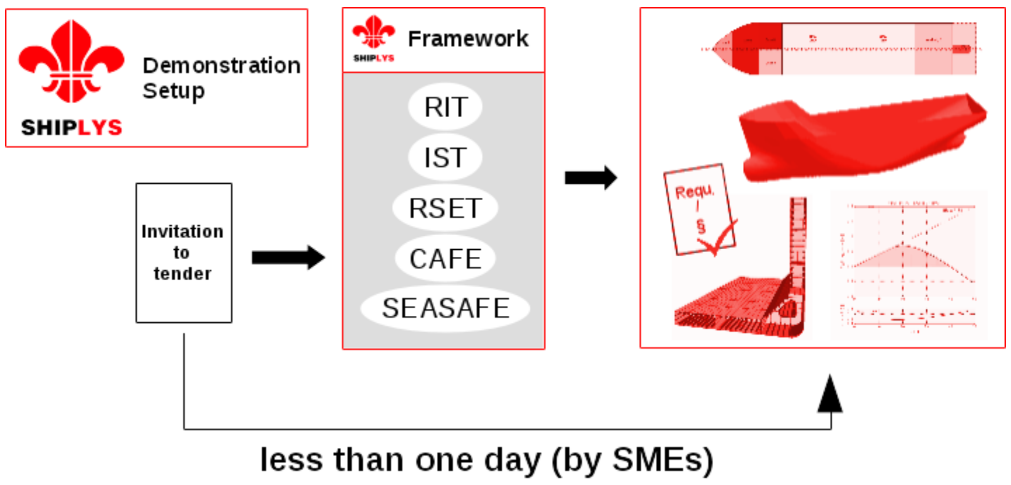

Fig. 7: Prototype scope

8. Conclusions and Outlook

The framework development has shown to be an interesting approach to make use of technologies evolving from the next generation of distributed systems. At the same time, these information technology-oriented mechanisms have been applied in a very concrete application-oriented context for early ship design methodology. This results in some interesting features and capabilities:

- Definition and maintenance of a tool set (or tool chain) becomes fully transparent and manageable, that making the working environment fully configurable,
- Access to tools is provided in a consistent way,
- Due to the distributed nature of the framework, the working environment is scalable. For example, computational intensive applications can be provisioned on computing nodes. Depending on the actual needs this can be realised either on moderately sized local CPU resources or by remote facilities.

With the framework being operational, next steps will be for example: to create a design process monitoring tool that is capable of documenting the progress of early ship design using the underlying process model, to integrate a wider selection of tools and to realise additional application scenarios (which have already been prepared in the project) – including a retrofitting case.

Future development could also put additional focus on the possible improvements with regard to communication among additional partners such as suppliers or subcontractors cooperating in a design project.

Acknowledgements
This paper presents work done in the project "Ship Lifecycle Software Solutions" (SHIPLYS) that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 690770. It reflects only the author's views and European Union is not liable for any use that may be made of the information contained therein. The authors also wish to thank the partners of the SHIPLYS Project for their input and support.

References

AES (2018), Topgallant Information Server, http://www.atlantec-es.com/topgallant-product-is.html, (last accessed 2018-03-16)

AFWAL (1981), ICAM Architecture Part II-Volume IV - Function Modeling Manual (IDEF0), AFWAL-TR-81-4023, Ohio: Materials Laboratory, Air Force Wright Aeronautical Laboratories, Air Force Systems Command, Wright-Patterson Air Force Base.

BHARADWADJ, U. (2017), Ship Lifecycle Software Solutions (SHIPLYS) – an overview of the first phase of development and challenges addressed, Proc. of Annual Conference of International Maritime Association of the Mediterranean (IMAM 2017, Lisbon), Rotterdam: A.A. Balkema Publishers.

BIM (2011), Lu, W.W.S.; Li, H., Building information modeling and changing construction practices, Automation in Construction, Vol. 20, No. 2, pp.99–100

BRANDES, U, M., et al. (2002), GraphML Progress Report: Structural Layer Proposal. Proc. 9th Intl. Symp. Graph Drawing (GD '01), LNCS 2265, pp. 501-512, Berlin: Springer-Verlag, 2002.

CAFE (2011), Bralić, S., Frank, D., Klanac, A., Cace, B., CAFE: An Innovative Multi-User System for Rapid Generation of Ship Concepts, COMPIT Conference, Berlin, German

DOCKER (2017), Docker Inc, What is docker?, https://www.docker.com/what-docker, (last accessed

2018-03-16)

FIELDING, R. T. (2000), Architectural Styles and the Design of Network-based Software Architectures, Dissertation, University of California, Irvine, www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation_2up.pdf,(last accessed 2018-03-16)

HCM (2016), OpenHCM Consortium: OpenHCM Description, https://openhcmstandard.github.io/, (last accessed 2018-03-16)

IDEF (2018), IDEF0 Function Modelling method – Strengths and Weaknesses, http://www.idef.com/idefo-function_modeling_method/ (last accessed 2018-03-16), College Station: Knowledge Based Systems, Inc.

ISO (1994), ISO 10303-1, 1994: Industrial automation systems and integration - Product data representation and exchange - Part 1: Overview and fundamental principles. Geneva.

ISO (2001), ISO 10303-214, 2001. Industrial automation systems and integration - Product data representation and exchange - Part 214: Core data for automotive mechanical design processes. Geneva.

ISO (2011), ISO 10303-203, 2011. Industrial automation systems and integration - Product data representation and exchange - Part 203: Configuration controlled 3D design of mechanical parts and assemblies. Geneva.

ISO (2003), ISO 10303-216, 2003. Industrial automation systems and integration - Product data representation and exchange - Part 216: Application protocol: Ship Moulded Forms. Geneva.

ISO (2004), ISO 10303-215, 2004. Industrial automation systems and integration - Product data representation and exchange - Part 215: Application protocol: Ship Arrangement. Geneva.

ISO (2004a), ISO 10303-218 2004. Industrial automation systems and integration - Product data representation and exchange - Part 218: Application protocol: Ship Structures. Geneva.

ISO (2005), ISO 10303-227 2005. Industrial automation systems and integration - Product data representation and exchange - Part 227: Application protocol: Plant spatial configuration. Geneva.

ISO (2013), ISO 16739 2013: Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries, Geneva.

JSON (2017), ECMA Int'l: ECMA-404 JSON Data Interchange Format, https://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf, (last accessed 2018-03-16)

KOCH, T.; KREUTZER, K. (2017), Refactoring Early Ship Design Methodologies, IMAM 2017 (International Maritime Association of the Mediterranean); Lisbon, Portugal 2017.

OPENSTACK (2017), OpenStack, What is open OpenStack?, https://www.openstack.org/software, (last accessed 2018-03-16)

RFC (2008), RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2, https://tools.ietf.org/html/rfc5246, (last accessed 2018-03-16)

RFC (2012), RFC 6749, The OAuth 2.0 Authorization Framework, https://tools.ietf.org/html/rfc6749, (last accessed 2018-03-16)

RFC (2014), RFC 7230-7237, Hypertext Transfer Protocol HTTP/1.1, https://www.w3.org/Protocols/rfc2616/rfc2616.html, (last accessed 2018-03-16)

RSET (2013), Depetro, A..; Hoey, R., Rapid Generation and Optimisation of Ship Compartment Configuration based on Life Cycle Cost and Operational Effectiveness, https://www.bmtdesign technology.com.au/media/4752073/optimisation_of_ship_compartment_configuration.pdf, (last accessed 2018-03-16)

S1000D (2016): ASD, AIA, ATA: International specification for technical publications using a common source database Issue No. 4.2,
http://public.s1000d.org/Downloads/Pages/S1000D/Downloads.aspx, (last accessed 2018-03-16)

SEASAFE (2017), About SEASAFE Professional naval architect package, http://www.seasafe software.com/webpage/htm/about_seasafe_package.htm, (last accessed 2018-03-16)

SHIPLYSD31 (2017), Koch, T.; Randall, G.; Hoey, R.: D3.1 Existing prototyping models and approaches in shipping and other industry sectors, http://www.shiplys.com/library/deliverables/d31-existing-prototyping-models-and-approaches-in-shipping-and-other-industry-sectors, (last accessed 2018-03-16)

SHIPLYSD33 (2017), Koch, T.; Randall, G.; Hoey, R.: D3.3 Requirements for the Integration of SHIPLYS tools and compatibility with existing tools,
http://www.shiplys.com/library/deliverables/d33-requirements-for-the-integration-of-shiplys-tools-and-compatibility-with-existing-tools/, (last accessed 2018-03-16)

VIRTUAL (2006), Vmware Inc, Virtualization Overview,
https://www.vmware.com/pdf/virtualization.pdf, (last accessed 2018-03-16)

YWORKS (2017), yEd Graph editor, https://www.yworks.com/products/yed, (last accessed 2018-03-16)