|  |  |
| --- | --- |
| From: ENAV | ARM10-3.1.1 |
| To: ARM and VTS Committees | 5 April 2019 |

Working Document

A lay-person’s Description of e-Navigation

e-Navigation is based on the same principles of data exchange as used in the apps that run on smart phones. For example, a weather app provides information to a user, this could be called a “Weather Service”.

This weather service satisfies the user’s need for information about the weather. Similarly, a ***Maritime Service***, in the context of e-Navigation, satisfies a user need for information concerning vessel navigation and other maritime considerations including safety, efficiency and the protection of the marine environment. Our weather service app has to communicate with a server that runs software that can provide weather information.



Figure 1 – The Concept of a Client-Server based Maritime Service

The interaction between the app and the server is defined by a ***technical service specification***, which describes the exchange of standardised messages and the language that is used in the message contents. The language is described by a ***data model***.

The software running on the server hardware is described as being an ***instance******of the******technical service***.



Figure 2 – The relationship between specifications of Maritime Services, Technical Services and data models in e-Navigation

If somebody wants to develop an app that is able to communicate with the server, or develop a server to be used by the app, then the developer must refer to the technical service specification and the referenced data models.

Similarly, an e-Navigation Maritime Service draws together information through a combination of one or more running ***instances*** of a ***technical services***. A technical service facilitates the exchange of data, by receiving messages, processing the data, and sending a result. The data is formatted according to the Common Maritime Data Structure, which is based on the IHO S-100 series of ***data models***, thus ensuring harmonisation and interoperability.



Figure 3 - The dataflow between Maritime Services, Technical Services and data exchange

We can further explore the relationship between Maritime Services, Technical Services and Data Models with reference to Figure 3, which further expands on our example of a Weather Service App.

Here we now assume that our weather service app is now part of a set of e-Navigation Maritime Services, and parallels can be made with Figure 2. The Maritime Weather Service App developed by StormyCorp Ltd. is composed of several Technical Services. Each Technical Service exchanges messages with servers that contain various data about characteristics of the weather. In this example we are using Temperature, Wind and Cloud Cover data. These datasets are obtained from sensors distributed around a geographical region and stored in databases. It is the job of the technical services to extract relevant data depending on the requirements entered into the user interface of the Weather Service app. The Weather Service app connects to an instance of Technical Service 1 for Temperature, Technical Service 2 for Wind, and Technical Service 3 for Cloud Cover (or Sunlight).

Let us examine the characteristics of the architecture outlined in Figure 3:

1. ***A single Maritime Service can use several Technical Services.*** This is an example of a 1:N relationship between Maritime Services and Technical Services.
2. ***The instances of these Technical Services can be deployed by a single organisation OR by multiple organisations.*** In our case for example, BlowFelt Inc. host a Temperature Technical Service on their servers and provide their Temperature Data. BlowFelt Inc. also provide a Wind Technical Service and provide their own Wind database. But the StormyCorp Ltd. Weather Service App has chosen to use Cloud Cover data from GreySkies Inc. BlowFelt and GreySkies servers could be widely separate.
3. ***Instances of Technical Services can be re-used by other Maritime Services.*** In Figure 3 we see a Maritime Service that provides a prediction of the amount of solar energy that can be harvested; the Solar Energy Predictor App developed by a company called SunShine Inc. A user of the app might enter a geographical location into the user interface to gain an understanding of the energy harvesting capability of his remote solar powered autonomous ship at its current location. Notice that SunShine’s App could use the same instance of GreySkies’ Cloud Cover Technical Service as used by StormyCorp’s Weather App, or it COULD have used the one hosted by BlueSkies. Decisions on which service provider’s technical service instance to use could depend on perceived quality of provided data, cost of subscription or numerous other factors.