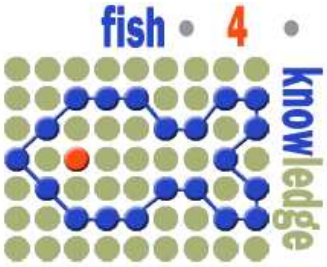


Fish4Knowledge Deliverable D7.4

Year 1 annual report to EC - Core

Principal Author: UEDIN
Contributors: All partners
Dissemination: PU

1 Project Objectives for Year 1

The main objectives of project year 1 were:

- To get the data capture and storage process working on a regular basis.
- To develop the fish detection and tracking basic algorithms (although enhancements are expected).
- To acquire initial data and ground truth for the fish recognition.
- To design the content for the database and some plausible structures for its distributed storage.
- To construct the system architecture for the data processing and delivery.
- To identify the questions that marine biologists would like to address with this sort of data.
- To design an interface that would enable the marine biologists to ask the questions.
- To design the ontological structures that support the information analysis and query answering.
- To design the bulk data processing workflow, and a software system architecture that enables it.

We have achieved all of these, as can be seen in the workpackage details below.

2 Work Progress and Achievements during the Period

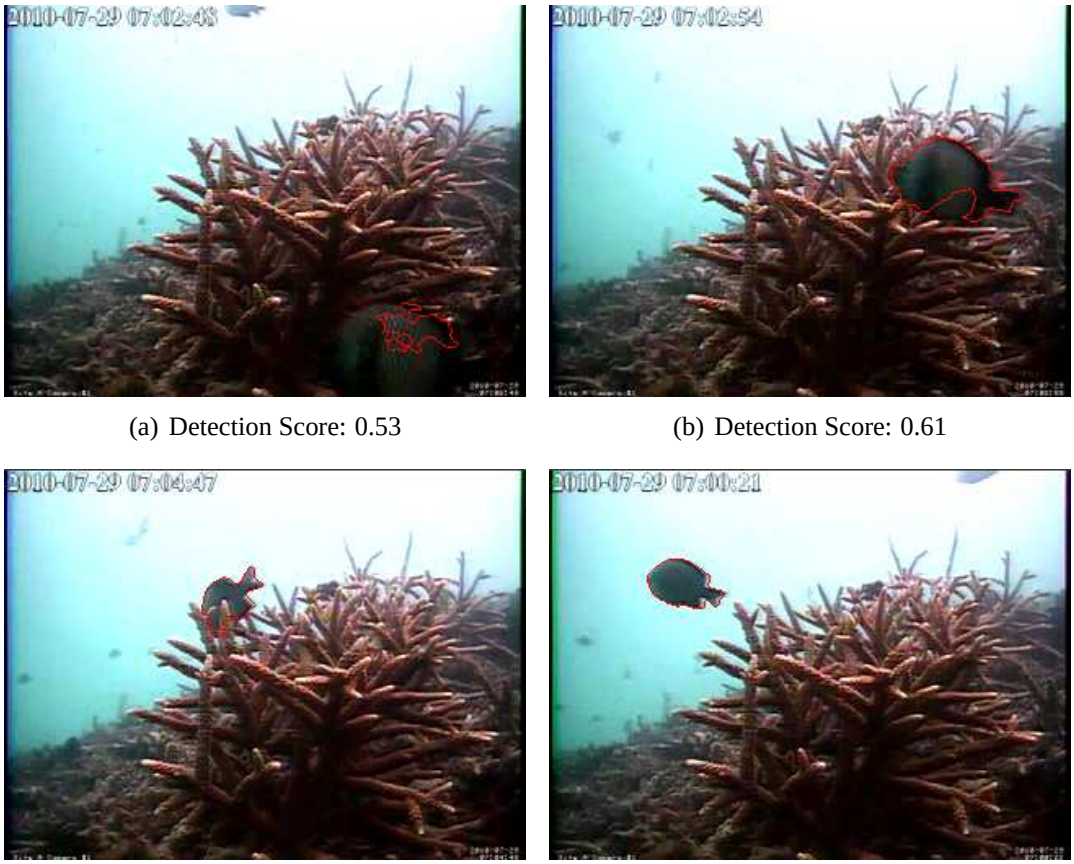
2.1 WP 1: Video Data Analysis

The aim of workpackage WP1 is fish/marine animal detection, tracking and recognition and clustering of unrecognised fish. WP1 provides the basic evidence for higher-level analysis: behaviour understanding, event detection, population statistics generation, workflow composition and more in general, it is the basis of all the answers to the 20 marine biologist questions. WP1 involves four main tasks: 1) fish detection; 2) fish tracking; 3) fish description and 4) fish recognition and clustering. However, the specific application underwater context makes these tasks very challenging: underwater video capture constrains the quality of the video (because of the technical difficulties in the underwater environment and in linking the cameras to mainland servers) and the targets themselves (i.e. fish) have more degrees of freedom in motion than, for example, people or vehicles in urban environments.

2.1.1 T1.1 - Fish detection

To deal with the various environmental difficulties found in underwater videos (such as light changes, murky water, waving plants), four fish detection algorithms have been implemented and tested for dealing with the presence of periodic and multimodal backgrounds, illumination variations and arbitrary changes in the observed scene: the well-known Gaussian mixture model; a mixture model variant based on Poisson distributions; an approach based on intrinsic images to deal with illumination changes; a frequency-decomposition technique to handle periodic movements of background elements, such as plants. These algorithms rely on a background-modelling approach, which consists in estimating a "background image" (i.e. how the scene looks like without fish) and comparing each new frame to this model, in order to detect pixels which deviate from it and which are marked as foreground. Of course, since the environmental conditions are likely to change with time, model update strategies have been implemented to keep the background description up-to-date with the current scene.
The reason for implementing and testing several detection algorithms is to assess the suitability of each of them to the different scene conditions in order to provide the higher processing levels (e.g. the workflow composition level) with different alternatives to use for answering user queries. The detection performance has been improved by adding a post-processing filtering stage, in which objects are selected according to a confidence level describing how sure we are that a detected blob is a fish. The computing of this score is based on the analysis of the colour and motion vector in the proximity of an object's contour (to check whether there is a marked difference between object and background) and inside the object itself (to check for uniformity of motion and colour). A few examples of detection scores are shown in Fig. 1. The inclusion of SIFT key points, extracted from fish clusters, into the computing of the detection confidence level is under investigation so as to reduce the number of false positives.
Although such methods have demonstrated good performance in detecting fish, especially when combined with the post-processing filtering stage (on average, a detection rate of 80% and a false alarm rate of 10% against high quality hand-labelled ground truth), they still present some drawbacks mainly due to the multimodality of the background that cannot be modelled using a specific probability density function, as deviations from the assumed model are ubiquitous. For this reason we are now implementing an approach that does not opt for a particular form of the probability distribution function (pdf), where each background pixel is modelled by a mixture of arbitrary pdf, whose distribution is identified while new pixel values appear, and by a set of samples of values rather than with an explicit pixel model.

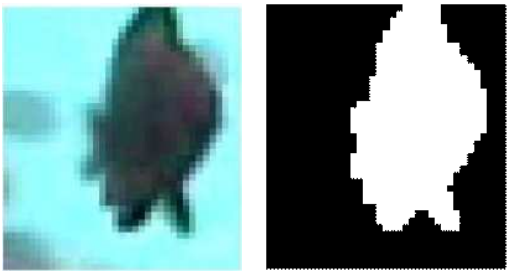
One crucial aspect of the fish detection task is the quality of the extracted contours, and the ways how to enhance such contours. The quality of the extracted contours was estimated by calculating the number of pixel-wise true positives (pixels correctly detected as belonging to the fish), false positives (background pixels detected as part of the object) and false negatives (object pixels mistaken for background) for each fish correctly identified by a detection algorithm. The average pixel detection rate (PDR) was about 90% whereas the average pixel false alarm rate (PFAR) was about 18%. The enhancement of fish contour quality has been carried out by applying image segmentation, which is a quite challenging task in the environments we are dealing with. We tested common approaches based on region growing, watershed and unsupervised k-means achieving promising results when high contrast fish/background occurs (see an example in Fig. 2), while they fail in cases with low contrast (an example in Fig. 3).



(a) Detection Score: 0.53 (b) Detection Score: 0.61

(c) Detection Score: 0.75 (d) Detection Score: 0.89

Figure 1: Examples of achieved detection scores.



(a) (b)

Figure 2: a) Original Image with high contrast fish/background, b) Output of the detection task enhanced with segmentation carried out with the simple region growing algorithm.

(a) (b)

Figure 3: a) Original Image with low contrast fish/background, b) Output of the detection task enhanced with segmentation carried out with the simple region growing algorithm

To deal with such cases, self-organising maps that exploit colour values, motion, temporal and spatial information are in development. Moreover, we plan to use also prior knowledge in the form of templates derived from fish clusters for accurate segmentation.

At the current stage, the detection performance is satisfying and much of the efforts for improving performance are due to the low quality of the initial set of videos. Therefore, we expect to achieve better results once videos with higher quality (higher native spatial resolution or improved resolution using super-resolution approaches) will be available or once enhancement techniques (contrast stretching, colour equalisation, etc.) will be added in a preprocessing stage.

2.1.2 T1.2 - Fish tracking

Fish tracking is necessary to consistently count the number of unique fish in the video (which of course is less than the number of total appearances) and to provide data for the following stage of event detection and behaviour understanding based on trajectory analysis. The tracking algorithm chosen to handle all the phenomena typical of the underwater domain exploits covariance matrices to describe the appearance of the objects in the scene. We adopted a novel covariance representation that models objects as the covariance matrices of a set of features built out of each pixel belonging to the object's region. In detail, for each detected object, the corresponding covariance matrix is computed by building a feature vector for each pixel, made up of the pixel coordinates, the RGB and hue values and the mean and standard deviation of the histogram of a 5×5 window with the target pixel as centre. The covariance matrix, which

models the object, is then computed from this feature vector and associated to the detected object. Afterwards, this matrix is used to compare the object with the currently tracked objects, in order to decide which one it resembles the most.
This representation takes into account both the spatial and statistical properties, unlike histogram representations (which disregard the structural arrangement of pixels) and appearance models (which ignore statistical properties). The results obtained with this algorithm show that it can accurately follow a fish even when it is temporarily hidden behind plants or in the presence of similar fish in the scene. However, the accuracy of the algorithm is strongly linked to that of the detection algorithm, since it assumes that all and only moving objects will be provided by the underlying object detection algorithm; for this reason, tracking may fail because of detection inaccuracy.
Given the importance of the tracking component in the overall project, due also to the fact that it is responsible for identifying the fish trajectories on which the behaviour understanding and event detection modules will rely, we have integrated in the system an approach for the assessment of the quality of each extracted trajectory, in order to filter them and select the ones that respect specific criteria of goodness, thus avoiding to invalidate the higher level analyses. To compute such a quality score, we have adopted a series of measurements on appearance, texture and motion in order to obtain a value indicating the goodness and the plausibility of each tracking decision and, more in general, of a trajectory. Fig. 4 shows a few sample trajectories with related average quality scores (computed as the average of the scores for each tracking decision of a trajectory); it is possible to notice that the trajectory of the top-left image is unrealistic (and is caused by a temporary mis-association between objects) and its average score, computed as average of the scores of each tracking decision for all the appearances of a fish (four times, in this case), was 0.63, whereas the image on the top-right side shows a correct trajectory whose average score is of 0.91. The two bottom images show, from left to right, a complex but correct trajectory (with a 0.81 score) and a trajectory which is correct up to a certain point, before an occlusion happened, so its total tracking score is 0.71.

To evaluate the performance of the covariance-based tracker, we have compared the results obtained by the application to a set of manually-labelled ground-truth videos. Moreover, since ground-truth generation cannot be performed on a large number of videos, as it is a very time-consuming and error-prone operation, we also integrated an automatic evaluation framework, based on the above-described tracking quality scores, into the system. This framework allowed us to test tracking algorithms without resorting to manual generated ground truth data.
Our results show that the developed covariance-based algorithm is able to correctly identify 91.3% of the fish on set of five videos with hand-labelled ground truth; as for the trajectory quality, the average correspondence between algorithm-computed trajectories and ground-truth trajectories is 95.0%, and the correct decision rate is 96.7%. These values are about 10% higher than those obtained by the CAMSHIFT algorithm, the only approach previously tested on underwater video clips. Then we also tested the accuracy of our tracking algorithm on a set of about 3000 unlabelled videos using our on-line framework, achieving an average accuracy of about 84% in tracking fish, whereas CAMSHIFT obtained an average accuracy of 73%. Since the computed quality score has demonstrated to be a reliable measure for assessing the quality of each extracted trajectory, we are currently working on formulating tracking as an optimisation problem where the global maximum score has to be found in consecutive tracking decisions for each trajectory. This would allow us also to repair tracking failures.

(a) Unusual fish trajectory

(b) Plausible fish trajectory

(c) Plausible fish trajectory

(d) Fish-plant Occlusion

Figure 4: (a) an erroneous path (average quality score 0.63) due to a failure of the tracking algorithm; (b) a correct path (average quality score is of 0.91); (c) a complex but correct fish path with an average average quality of 0.81; (d) trajectory with an average quality score of 0.71 due to a fish-plant occlusion.

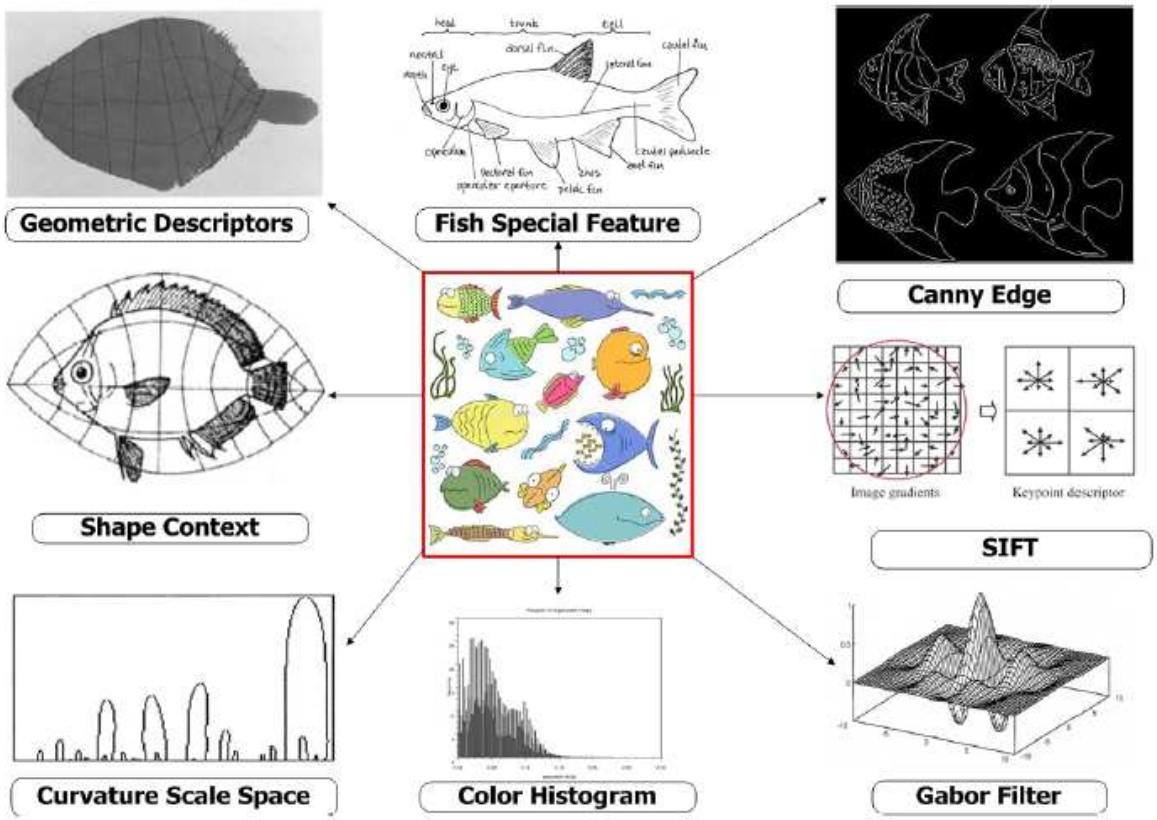


Figure 5: Several higher and lower level fish descriptions which can be used for recognition

2.1.3 T1.3 - Fish description

Fish description is necessary for fish detection, recognition and clustering as shown in Fig. 5, which depicts some descriptors that can be used for fish recognition and clustering (Figure 5 is from the PhD thesis proposal of Xuan Huang who works on fish recognition).

Beyond the fish properties already used in the implemented approaches, the computer vision groups of UCATANIA (UC) and UEDIN (UE) agreed on a more refined list of descriptors (some already available and some others to be developed) that will be used in both fish detection and fish recognition process. Most of these descriptors are affine invariants, since they have to deal with the different position, orientation and scale of fish with respect to the camera. Table 1 shows this list and the categorisation (colour, texture, motion and contour) of the descriptors we intend to use. For each fish description, we also mention which partner (UC, UE or both) is responsible according to task relevancy, i.e. all the features assigned to UCATANIA will be used likely in the detection and tracking tasks for improving performance, whereas the ones assigned to UEDIN will be used in the recognition and classification tasks.

Invariance of these descriptors against types of changes (e.g. light intensity change) and types of transformation is under investigation. This list will be added to a wiki page where each descriptor is described in detail specifying data structure and range. This allows non-experts to get an idea what the description does and gives experts a handle how to use the fish descriptions in their code or from the database.

Colour		Texture		Motion		Contour	
Name	**Resp.**	**Name**	**Resp.**	**Name**	**Resp.**	**Name**	**Resp.**
Background Scoring *	UC	Gabor Filter	UC/UE	Motion Vector	UC	Rigid Points *	UC
RGB, nor RGB	UE	SIFT	UC/UE	FTLE	UC	Curvature Scale Space	UC
HSV, HSL	UE	SIFT with Global Context	UE	Periodic Motion		Curvature Points	UC
L_{ab}	UC/UE	PCA-SIFT	UE	Analysis *	UC	Fourier Descriptors	UC
Joint Histogram	UC	Covariance Matrix	UC			TPS	UE
Transformed Colour *	UC	Co-occurences Matrix	UC			ASM/AAM	UE
Colour Moments	UC	Spots/Stripes	UE			MDL	UE
HSV SIFT *	UC	Symmetry Hierarchies *	UC			Shock Graph	UE
RGB SIFT *	UC					Mellin Transform	UE
						Wavelet Transform	UC
						Implicit Polynomials *	UC

Table 1: Preliminary List of Fish Descriptors that will be used in detection, tracking and recognition processes. Most of these descriptors have been already implemented except the ones indicated with *.

2.1.4 T1.4 - Fish recognition and clustering

The work on the fish recognition and clustering components has started. At the start, several problems have been observed, namely uncontrolled environments and a large unannotated dataset of fishes. To deal with the uncontrolled environment, robust and invariant fish recognition methods need to be developed. Several methods are in development to determine the direction in which the fish is swimming, allowing us to determine the head and tail of the fish. This information can be combined with the tracking to obtain even more accurate results. In Figure 6, we show some preliminary recognition results on a small (495 fish) already labelled database. There are 11 species and each species has 45 images. 10-fold cross validation has been applied. The results are very promising with 96.9% accuracy, however larger datasets are necessary. This is because: 1) our initial groundtruth contained many observations of the same tracked fish, so recognition was much easier, 2) we only have a small amount of variation of individuals, environments, etc, 3) this is only for the 11 species most often seen (out f about 1000-1500) in the initial groundtruth dataset obtained from about 3000 detections.

This brings us to the second challenge which is the large amount of unannotated fish data. We developed a fish clustering method which is able to group similar fish together. This method is unsupervised having the advantage that it is able to discover new combinations of fish descriptions, allowing us to search for new fish species in the dataset. Because we have a large unannotated dataset, one of the challenges is to annotate a useful subset of this dataset in an efficient manner. We are working on an annotation framework where our clustering method supports users in the annotation tasks. This allows us to perform the annotations quickly while keeping high accuracy. At the moment, using this tool allows users to annotate fish in about 40% of the time it normally takes, with accuracy losses of around 3% which can be overcome by combining multiple users.

Generating ground truth for fish recognition

As discussed in Del5.3, a ground truth dataset needs to be created. A ground truth dataset is a set of images with annotations that are manually generated and checked for correctness, e.g., an image annotated with the names of the fish it contains. The ground truth serves two purpose. First, it can be used to evaluate the quality of software components. Second, it can be used

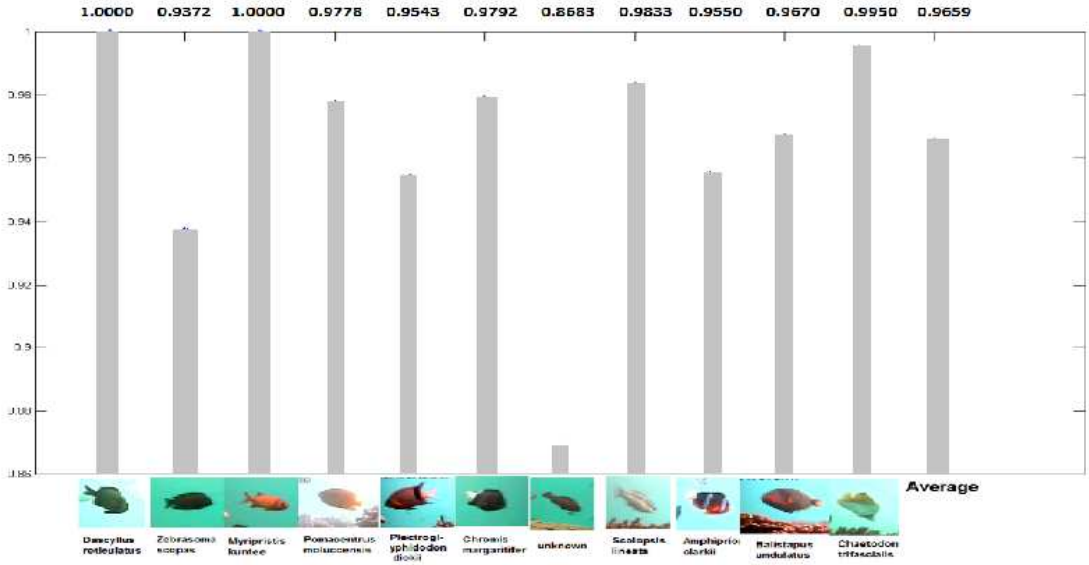
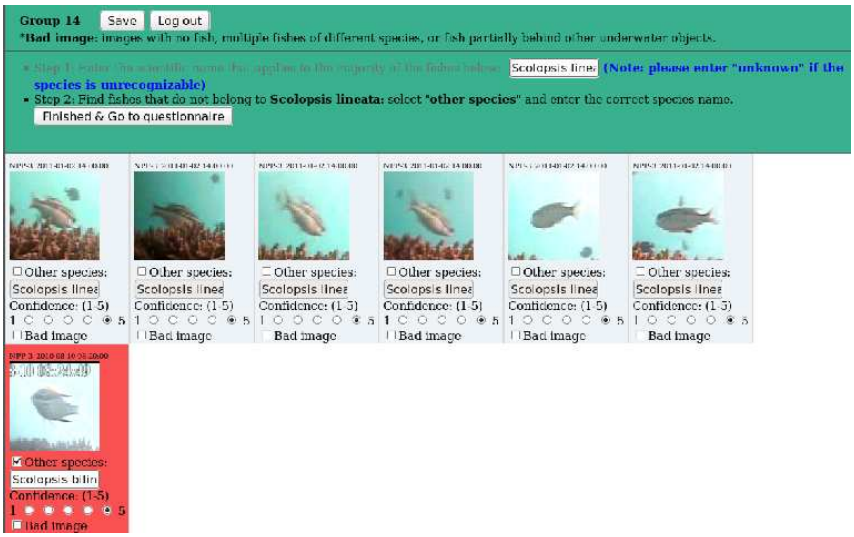


Figure 6: Plot of the latest fish recognition results on a dataset of 495 fish images of percentage recognition versus species. The final bin is averaged by species (not by fish, as some classes have many more examples).

as training data in machine learning contexts. Within the F4K project, a number of software components need such datasets, namely fish detection, tracking and recognition. We focused on creating the fish recognition ground truth. More specifically, the goal is to create an image set, in which each image is assigned with a scientific name of the fish it contains.

There were two challenges in this task: 1) in order to recognise fish with their scientific names, expert knowledge is required; and 2) manually annotating a large amount of images is tedious work. We separated the annotation task in two parts: expert annotation and the non-expert annotation, where UEDIN mainly focused on non-expert and CWI focused on the expert annotation. For both expert/non-expert annotation, a cluster-based approach is used to reduce the amount of work for the annotators. For non-expert annotation, the goal was to verify the fish clusters generated by the clustering algorithms. For expert annotation, actual species names need to be assigned to individual images.

To collect expert annotation, an interface was created to facilitate the annotation process:



The interface for expert annotators. The expert enters the species name for the majority of the images within a cluster, then selects images that should not belong to this cluster and inputs the correct species names for these images.

In short, instead of typing in a species name for each image, the interface enables the annotators to assign a species name to a cluster of fish images and automatically assigns the same name to all the images within the cluster. The annotators can then correct the individual names assigned to those images that do not belong to the same species within the cluster. In this manner, in the worst case, the annotator will have to manually assign a species name to each of the images, i.e., when the clustering is so bad that each image within a cluster contains a different fish species. In the best case, i.e., when the cluster is pure, the annotator only needs to enter the species name once.

In our experiment, we invited 3 marine biologists, who have over 10 years research experience in the area where the underwater video cameras are located. In order to obtain relatively high quality clusters, we manually constructed clusters over a small sub set of our dataset: 27 clusters over 524 images. Since the sizes of clusters are very imbalanced, for each cluster, we randomly sampled 30 images to be shown to the biologists. In total 190 images were annotated by the biologists. For 82.6% of the images, at least two biologists agreed on a species name; for 54.3% of the images, all biologists agreed on a name (including the case where two biologists agree on a species name while the third biologist indicate that he/she cannot identify the fish). A further examination shows that for some clusters the biologists do not agree on a name at the species level, but *do* agree on a name at a family or genus level. After finishing annotation, we also included a questionnaire for the biologists in order to collect information such as which features he/she used to identify certain species, why certain species are difficult to identify, etc. This type of information can be a useful hint for selecting useful features when developing automatic methods for fish recognition.

2.2 WP 2: Interactive User Query Interface

2.2.1 T2.1 - Establish user information needs

We interviewed potential users that are marine biology experts. We investigated 1) the most important queries they would ask the system to support, and 2) the context of use of the

requested information. We derived the needed domain-oriented information and the high-level user tasks.

Regarding user information need, we disclosed biology-specific measurements desired by end-users. We selected those that are feasible with F4K feature detection. We specified their calculation using the metadata available in the F4K database. User information needs, context of use, and biology-specific measurements are detailed in the Deliverable 2.1.

Regarding the high-level user tasks, we synthesised two concrete usage scenarios that expose the user needs, and the system functionalities that support them. We decomposed the scenarios into detailed tasks: we specified the manipulated data, and the functionalities for data manipulation. Usage scenario, high-level user tasks and detailed user tasks are described in the Deliverable 2.2.

A brief description of our findings states the following:

- User information needs are primary composed of measurements to describe biological facts as well as technical facts for assessing and trusting the validity video analysis.
- Secondarily, user information needs include environmental data collected through sensors or meteorological agencies, as well as user-defined environmental events.
- Biological facts that can be described with the F4K system primary concern population dynamics (e.g., demographical analysis) and impact of environmental conditions.
- Technical facts will be exposed to users by successively disclosing the underlying processes that produced the high-level information (e.g., biology-specific measurements), and the sets of videos that were automatically analysed.
- The high-level user tasks are: analyse population dynamics, analyse environmental impacts, verify analysis, personalise analysis.

Initial query performance analysis w.r.t. user requests

We conducted an initial analysis of the query performance analysis in terms of efficiency and effectiveness based on the 20 questions defined in D2.1. With this analysis, we aimed to have an initial idea of

- the efficiency of the current data storage in calculating users' request and the type of operations that might be time-consuming given the current storage;
- factors that influence the results of the data analysis/summarisation over the records in the fish database.

It was an empirical analysis using the fish database defined in D5.2 and at the moment of our experiment, it contained 337,343 fish and 2,809,581 fish detections. No fish species information was available. Given the information available in the database and the 20 questions, we focused our evaluation on 4 types of queries: 1) count fish over different time intervals; 2) count fish over different locations; 3) count fish over different locations and different time intervals; 4) count fish for a given location over given time intervals.

We conclude our results as follows. In terms of efficiency, we found that the efficiency of current data storage varies w.r.t. different types of queries. The number of records involved for

counting and the number of "join" operations over tables (e.g., in order to find the corresponding time and location information for a fish record) are the main factors influencing the processing speed. In terms of effectiveness, we found that missing data has an impact on the results of counting. Further, since the fish detection and tracking results come with a certainty score, it was expected that different ways of thresholding/combining the certainty scores will have an impact on the counting results. However, we found that if we only consider relative counts, e.g., the *change* of the amount of fish found per month over a year, setting a threshold on the certainty scores does not have an obvious impact.

2.2.2 T2.2 - Explore component-based prototypes

We plan to implement the following components:

- Metric Calculator: calculation of domain-oriented measurements using the metadata available in the F4K database and the environmental data.
- Query Engine: transformation of user queries performed through the graphical user interface into queries that can be computed by the Workflow and the Metric Calculator.
- Rendering Engine: managing the display of the web interface using state of the art rendering techniques (e.g., interactive diagram).

The Metric Calculator and the Query Engine will be developed gradually, to implement one by one the high-level tasks. The first implementation of the Rendering Engine will support the experimentation of various user interface prototypes, in order to research the user interface paradigms, and to collect feedback.

2.2.3 T2.3 - Support for high-level information needs

We plan to experiment with several user interface paradigms and interaction techniques. We identified 2 research directions:

- Multi-layered information access: how to allow users to access and manipulate the underlying computational processes? In addition to improving trust, can such multi-layered access also improve understandability and usability?
- Multi-faceted information access: how to support purposive information gathering, and flexible user-specific interpretations?

2.2.4 T2.4 - End-to-end system integration with data

No action on this task during year 1.

2.2.5 T2.5 - Evaluation and in situ user testing

No action on this task during year 1.

2.3 WP 3: Process composition and execution

Here we describe our initial efforts towards work contributing to WP3, the workflow component of the F4K project.

The workflow component of the F4K project is responsible for investigating relevant methodologies and implementing a working workflow system towards the end of the project. More specifically, its task is to take in video data that has been captured by the F4K project partner NARL and analyse and process them in useful ways to answer targeted user queries.

The approach that we have chosen for the workflow system is a knowledge-based one. This approach enables us to separate the problem and domain descriptions, the application computer vision and data analysis software components and the actual workflow enablement components, the workflow engine. These components are understood and connected via a set of knowledge based representations.

This loose-coupling approach enables us a more flexible mechanism that allows us to easier adapt to a different problem description (or indeed a different problem area when desirable). This is particularly useful, as the problem and domain descriptions may evolve over the life time of the F4K project. It also enables us to easier evolve and make use of new software components, as they will become available within the F4K project over time. It also enables us to improve the workflow enablement component, the workflow engine, to become more efficient and/or richer, as needed, as we make use of high performance computing facilities through NARL. This approach should therefore enable each of these major components of the system to evolve and improve independently, as needed, without needing substantial changes to other components. The main effort is to ensure the correct connections are in place between them.

The problem and domain descriptions and workflow engine are also written using knowledge-rich languages, i.e. in Prolog, that logic-based programming techniques can be applied directly. As a result, we are able to make use of the planning technologies that enable us to perform dynamic workflow composition and assist workflow execution.

The semantics of the problem and domain descriptions are defined in a set of ontologies that have corresponding knowledge-based representations. This provides semantically more human-understandable and uniform labels to annotate videos and images. Such labels are a translation of the results as produced by the image and video processing software components. These labelling will then be used to assist user-query answering functions that is a part of the workflow system.

In the sub-sections below, we describe in more details our efforts in creating the set of problem and domain ontologies as described above that will be used as input by the workflow engine. We also describe our initial efforts in our investigation and implementation for the workflow system enablement component, the workflow engine. We also report on further issues when engaging our workflow system with NARL's high performance computing facilities. An overview of the design of our workflow system that provides a connection of all of the above components is provided in Figure 7.

2.3.1 T3.1 - Create Domain Ontologies Based on User Requirements

The workflow component (see Figure 7 for overview) of the F4K project is responsible for the workflow composition and execution tasks, that takes in a set of videos it makes use of the video

image processing (VIP) modules and can run such tasks on high performance computing (HPC) machines, that is directed to address targeted user requirements.

As we have taken a knowledge-based approach for our workflow system, to address the various aims as presented to us, three types of knowledge are required as input for the workflow system:

- The user requirements: we need to understand and describe them in such a way that can be related to and therefore addressed by VIP tasks;
- The video data input: we need to analyse them and identify useful features in knowledge based descriptions;
- The VIP software modules that are made available to us: we need to relate and categorise their functions and describe their I/O requirements in knowledge based descriptions.

The first knowledge is acquired through correspondence with F4K's marine biologists in Taiwan, while the second and third knowledge are acquired from the image processing experts in F4K (Edinburgh University and University Catania teams). As a result, the workflow component interprets the user requirements (from User Interface team) as high level VIP tasks, create workflows based on the procedural constraints of the modules (provided by image analysis teams) to ultimately invoke and manage their execution in a distributed environment (HPC team).

We have created a set of suitable domain ontologies that are based on user requirements (from marine biologists) for our intelligent workflow system, as defined in the project proposal. The main purposes of these ontologies are (1) to support the development of appropriate functions of the workflow system, and (2) to serve as a communication media to interface with other Fish4Knowledge components. As a starting point, we described the 20 scientific questions as provided by the marine biologists. Then we provided an analysis of these 20 questions from the workflow system's point of view - that is based on the capabilities of video and image processing (VIP) modules available to us. Based on a mapping between the user requirements and a high level abstraction of the capabilities of the VIP modules, we have constructed the Goal Ontology. The Video Description Ontology contains the environmental factors related the videos. The Capability Ontology describes details of the VIP modules. To date, the Goal Ontology contains 52 classes, 85 instances and 1 property, the Video Description Ontology has 24 classes, 30 instances and 4 properties and the Capability Ontology has been populated with 42 classes, 71 instances and 2 properties.

The ontologies have been verified by F4K project partners and the relevant project deliverable (D3.1) has been completed and submitted. The ontologies were utilised in the first version of our workflow composition and execution system to support video classification, fish detection and counting tasks. The ontologies will continue to evolve with the projects needs and are envisaged to interface with other Fish4Knowledge components in due course.

2.3.2 T3.2 - Workflow System Design

The workflow component of this project interprets the user requirements as high level VIP tasks. It creates workflows based on the procedural constraints of the modules (provided by image processing experts) to ultimately invoke and manage their execution in a distributed environment.

There are two parts to the workflow design: 1) workflow composition; and 2) workflow scheduling and execution. Workflow composition deals with identifying the set video and image processing (VIP) modules to solve a user-provided VIP task. Workflow scheduling and execution deals with distributing the VIP modules onto high performance computing facilities and scheduling them optimally for our needs.

Workflow Composition

Workflow composition is the identification of a set of VIP modules that can solve a given VIP task based on user request. This component is designed using a planning and ontological approach within a three-layered framework, shown in Figure 7.

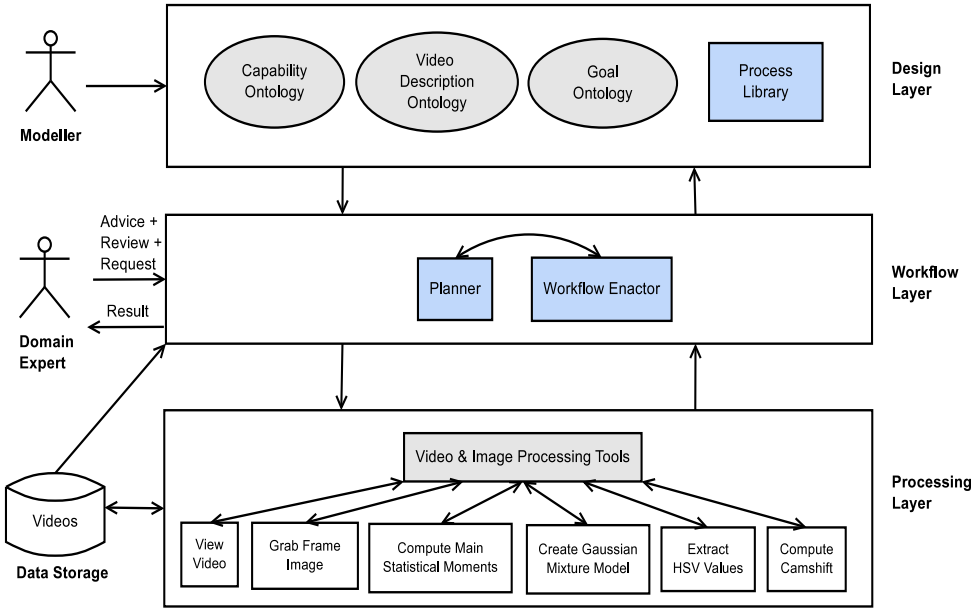


Figure 7: Overview of Workflow Composition Framework for Video Processing.

The design layer contains components that describe the VIP tasks, information about the video, image processing tools and processes to be carried out in the system. These are represented using the ontologies described in Section T3.1 and a process library. Knowledge about VIP tools, user-defined goals and domain descriptions are organised qualitatively and defined declaratively in this layer using these ontologies, allowing for versatility, rich representation and semantic interpretation. The process library contains the code for the primitive VIP tasks and methods available to the system. These are known as the process models. A primitive task is one that can be directly performed by a VIP tool, while a method is decomposed into primitive and non primitive tasks. The workflow layer is the main interface between the user and the system. It also acts as an intermediary between the design and processing layers. The workflow enactor ensures the smooth interaction between the components, access to and from various resources such as raw data, VIP toolset, and communication with the user. The main reasoning component is a planner that is responsible for transforming the high level user requests into low level video processing solutions.

The processing layer consists of a set of VIP tools that can perform various image processing functions. The functions of these tools are represented in the capability ontology in the design

layer. An initial version of this workflow composition and execution system on a single resource has been designed and evaluated. The next stage will involve mapping the workflows to multiple resources, and exploring efficient means to execute them in parallel.

Workflow Scheduling and Execution

The workflow scheduling and execution is dependent on the computing facilities provided by NARL, Taiwan. At present we are investigating means to distribute and schedule the VIP modules on a cluster made up of 96-CPU machine with two nodes, each containing 48 core. It is assumed that each node in the cluster will have a shared memory and a shared filesystem for the 48 processing units to work with. This will ease the parallelisation process.

We will explore ways to parallelise the execution of our VIP workflows using the cluster provided by NARL. In order to do this, we consider two criteria for a system's memory performance:

- Latency - the time it takes for the memory to process a request (*e.g.* a marine biologist's query)
- Throughput - the actual rate at which data can be pumped from the memory to the processor. This measure is proportional to bandwidth.

Throughput and latency will be used in conjunction with the types of queries that will concern us. It is anticipated that F4K will have two types of queries:

1. Off-line queries – these are queries that will be processed off-line and results stored in the database. When the user makes a query, the results are retrieved and aggregated from the database.
2. On-line queries – these are queries where results are not available in the database, hence they will have to be processed dynamically. Results will need to be computed via VIP workflow composition and execution before being presented to the user.

It is anticipated that most of the queries for F4K are going to be processed off-line, as the 20 questions posed by F4K's marine biologists would require the processing of large data (videos) using relatively large jobs (VIP modules). For such a scenario, a workflow management system that *maximises throughput* would be most appropriate. This means that large jobs can be run on large data optimally. Each VIP module can execute a relatively large task, such as fish detection and counting in a given video.

In the scenario of on-line processing, it is important that the user request is dealt with as quickly as possible, such that the time spent by the user waiting for results is minimised. In this case, the workflow management should be one that *minimises latency*. For this case, it would be advisable to break the existing independent VIP modules into smaller modules for more optimal parallelisation.

The next step is to combine a scheduling policy with a parallelism policy for executing the VIP workflows. Unless jobs are too short for execution, we will be adopting the **dynamic** workflow scheduling strategy for execution. This strategy will schedule jobs as resources become available. This is by far more efficient than scheduling statically whereby allocation of tasks is done only once and jobs are distributed to the processing units. Although dynamic

scheduling comes at the cost of scheduling time, it will be far more efficient for large jobs and large data. Finally we will investigate the performance of the workflow management system using the **task farm** parallelisation policy for our jobs. This would involve the execution of independent tasks (jobs) in multiple processing units, or processors. The tasks are split between the processors which work on them in parallel. As the calculations are independent, no information needs to be exchanged between workgroups during this time, and sharing of results can be postponed until all the tasks have completed.

On-going discussions with NARL will involve determining the exact specifications of the processors of the cluster along with their memory and filesystem configurations, selecting a scheduler, installing the VIP modules (C executables) and executing the off-line processing tasks for typical queries. User scenarios from the User Interface team would be required in order to determine what types of VIP tasks will need to be solved. This would enable us to compose, schedule and execute the workflows according to the user requests and evaluate the efficiencies of the parallelisation and scheduling techniques that we plan to deploy.

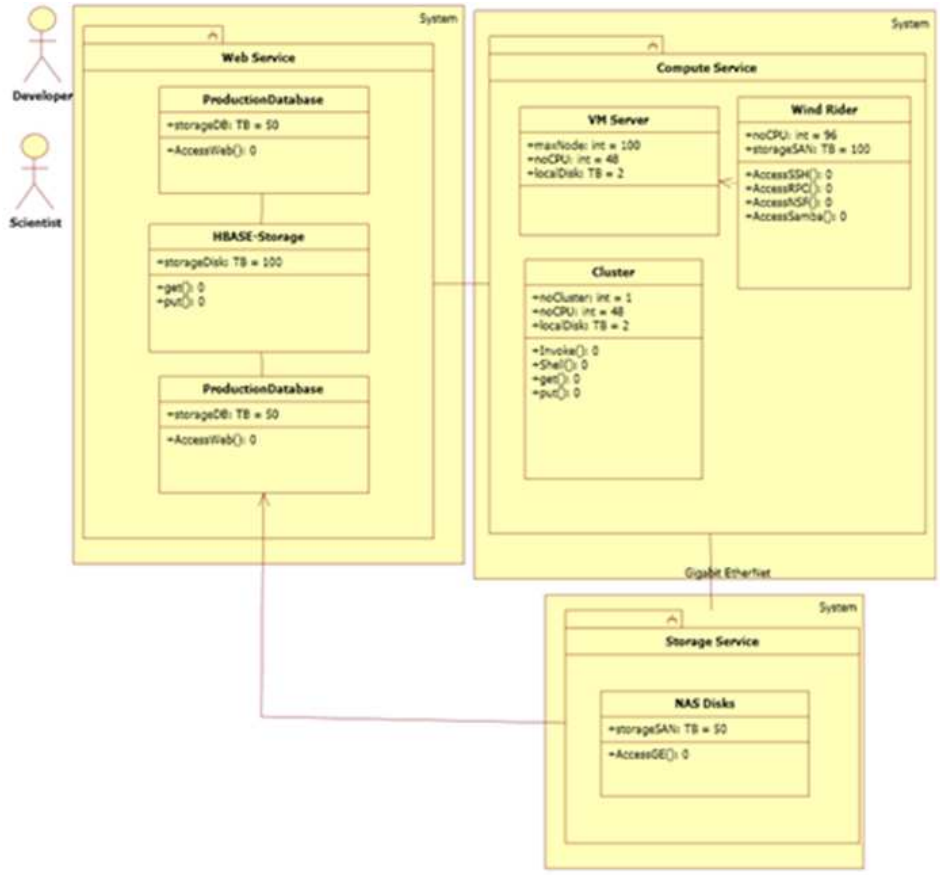
2.3.3 T3.3 - intelligent workflow system

No action on this task during year 1.

2.4 WP 4: High Performance Storage and Execution Architecture

NARL provides underlying infrastructure for the system of Fish4Knowledge, in which not only are the compute and storage resources prepared to meet the analysis and store demand for the daily streaming underwater observational video but also has the performance of the online advance query to be ensured for scientists.

In order to achieve such goals, NARL has built a 48 CPU multicore machine as a dedicated server for compute and storage services, coupling at the backend with the newly deployed supercomputer Wind Rider, in which 96 CPUs are dedicated to the system, and an array of NAS storage system with 10GE interface for fast data retrieval. On top of the system an execution environment is also developed to allow easy and effective use of the system by both developers and scientists. The high level architecture design is:



The system architecture design for high performance compute and storage services.

High performance multi-core processors at the observatory level do the first filtering and bulk compression of the raw video data before transmission across the network. Data and task level parallelism on cluster machines will be used to analyse the video to produce the RDF/XML store content, and execute the query filtering steps that extract content from the RDF/XML store.

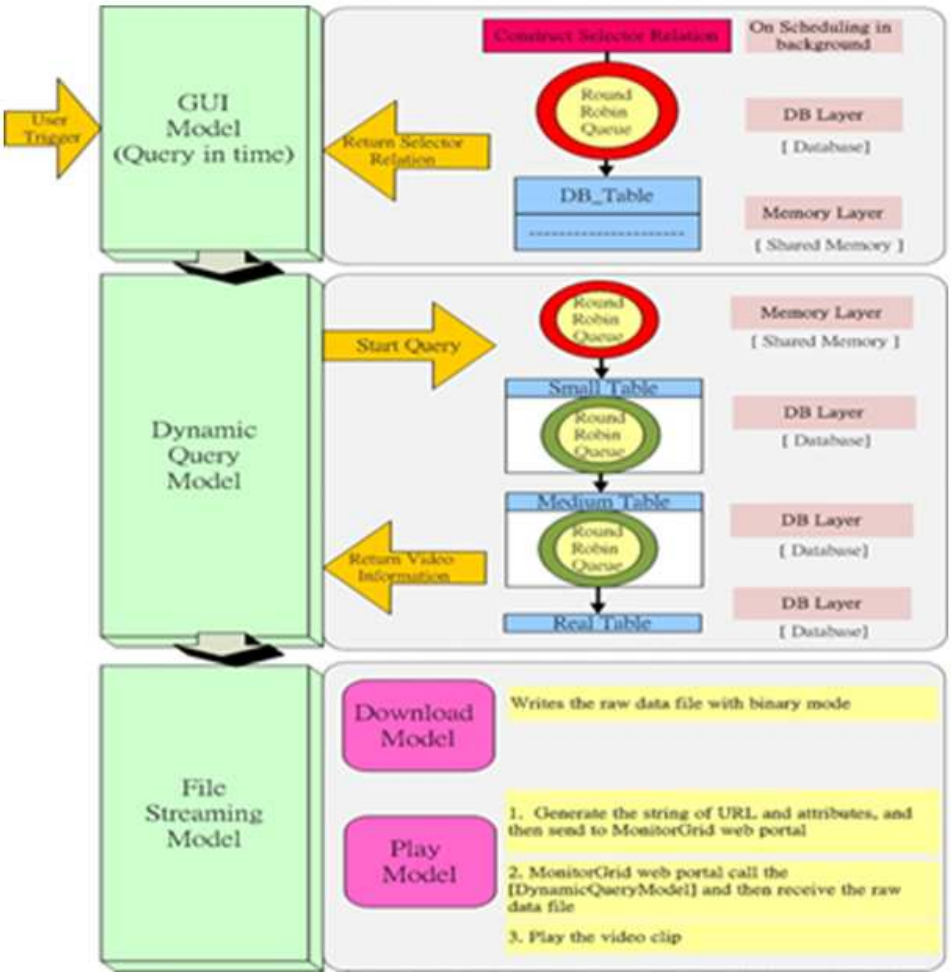
The current development is summarised as the following:

- 6 N7700 NAS, provide about 45 TB storage size
- 2 WindStar NAS, currently installed totally 16 TB storage, the storage size should be able to extended to maximum.
- Underwater camera, using HD camera and CCTV to provide eco-video as data source to the F4K system. The undersea camera infrastructure is provided from Ecology Plan of Water Resource Agency in Taiwan.
- 48 CPUs servers, provides virtual machine running platform. 96 CPUs from Wind Rider supercomputer are used for compute service.
- Video Query Portal, web-based interface for accessing recorded ecology video.
- Customised-based VM Portal - web-based interface to access a virtual machine in the system.

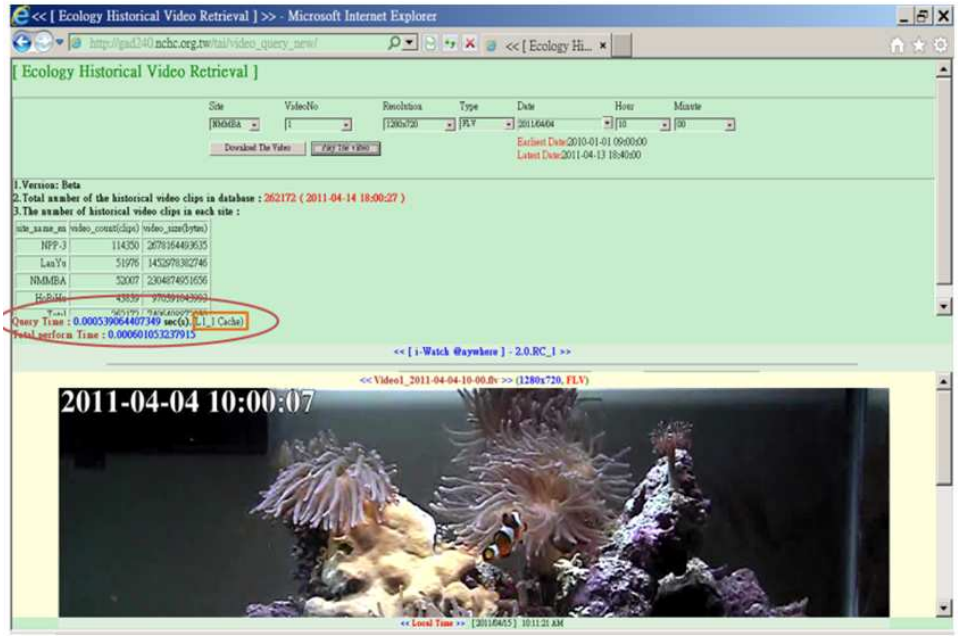
2.4.1 T4.1 Enhance current video capturing and storage

On the video capturing, NARL has incorporated 10 underwater camera in 3 different sites, Kenting, Hobihu and Lanyu. The sites are remote and only limited bandwidth of a public network available. Yet, higher resolution and higher frequency of the video are required for better analysis. Correspondingly, the amount of storage will dramatically increase. NARL increased the video frame rate from 8 fps to 24 fps and resolution to 640x480 for the underwater CCTVs, conducted sensitivity analysis for data compression technologies to search for an optimal one for the current network constrain, and scaled our NAS storage to satisfy the needs of the new setting. For the frontend, faster video query and display over the bulk video database are also required. The figures below show how the system is built and also the fast display portal that demonstrates a speed-up improvement.

Fast video query and display through the read-write manipulation in a pipeline of the first 4 tiers of 5 tier storage hierarchy, i.e. L1-L5:

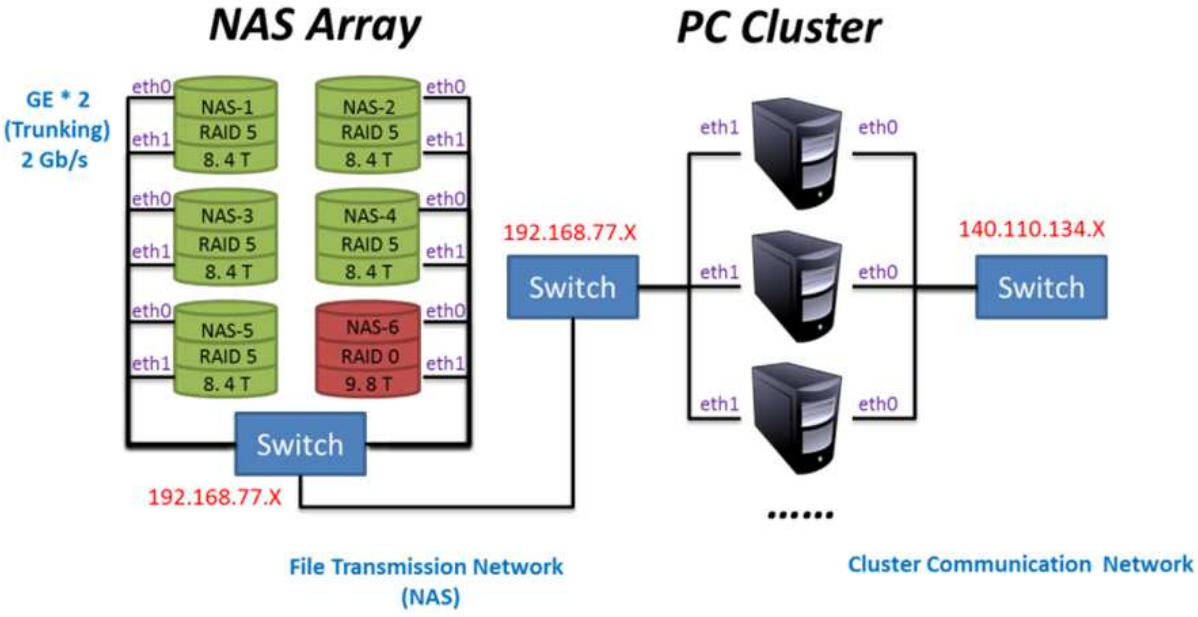


Video query portal shows the result at average speed-up 200% - 400%:



2.4.2 T4.2 Build data storage facility

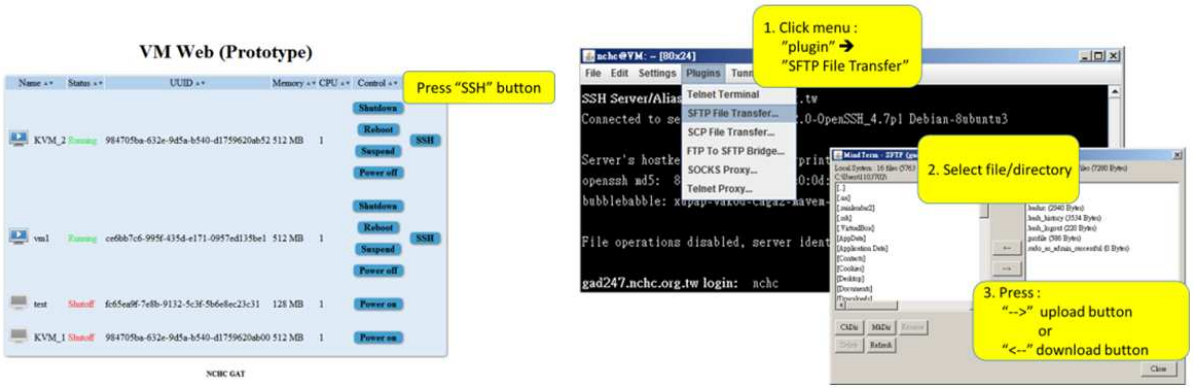
The F4K data storage facility is built partly via NAS storage and partly via NARL shared HPC storage facility, such as EMC CLARiiON CX700 High Efficiency Disk Storage Array and StorageTek L5500 Tape Silo. Total capacity is currently 60TB, and is growing. Here is the basic F4K storage built within NARL (NAS storage Array available for 1GE and 10 GE network interface):



2.4.3 T4.3 Develop process execution interfaces

NARL developed a virtualisation environment to tackle the complexity of the F4K development requirements and expect the virtualisation environment to be evolved into a production service

after the project. Such an environment will allow developers to conduct various tests as well as scientists to use a workflow, in which processes can be automatically composed and executed. A web-based VM (Virtual Machine) portal is developed as shown here:



2.4.4 T4.4 Develop distributed data and computational methods

In order to achieve high performance large scale data/query system, it is crucial to exploit some well-recognized performance enhancement approaches such as concurrency, locality (e.g. local caching or in situ computing), multi-core computing and GPU computing for both data management and data/video analysis: NARL built an F4K infrastructure by using advanced multi-core compute facility coupling with large NAS storage arrays and optical switches for connectivity. An automatic workflow system will be used to orchestrate distributed processes of data/video analysis and queries, which will result in sizeable concurrent threads running in the system. How to ensure the system being highly available and reliable strongly depends on the quality and level of the concurrency that can be achieved. Current efforts are on porting distributed and automatic workflow system to the infrastructure and evaluate its collective performance.

Previous works in access remote video clips take advantage of hierarchical memory with local caches to reduce the frequency of data loading from remote sites, which proved to be useful for streaming data. It is not clear if this model can still be as effective when the data size increases to Tera-scale or even Peta-scale. In addition, the videos being streamed from the underwater video cameras at the field station to the NARL data repository are constrained by the bandwidth of the last mile network. In situ computing is necessary either for early filtering of video or content-based compression, which brings intelligence to the local so that useful information can be extracted in advance and redundant information removed. As a result, the data size will be further reduced to meet the network constrain, but the quality of the data still remains.

Multicore technology is used extensively in the mainstream HPC system architecture. As mentioned earlier, NARL compute infrastructure is built based on such technology. Many variants of MPIs being optimized for multicore were developed in HPC community. NARL will support the optimization of such MPI-based parallelism in our infrastructure. GPU and GPU/CPU computing can accelerate codes of some applications to an extreme that cannot be done otherwise. GPU infrastructure and implementation based on CUDA will be supported in the final phase of the project.

The above approaches will be also used in database development in the project. To tackle

with large data set, there are three database technologies that are strongly relevant to us, which are data warehousing, distributed database and NoSQL. Currently MySQL and PostgresSQL are implemented for practice. The performance of parallel and distributed versions of both databases is being tested. Further research will be conducted for the above database technologies to meet the project goals in the presence of Tera-scale video data.

Database distribution

We have been analysing the expected total data load based on our current estimates of data capture and analysis. We have also looked at the sorts of questions likely to be asked of the system by potential marine biologists. From these we have designed a hypothetical distributed data system suitable for answering questions in a few seconds, even over the full dataset. Details of this follow.

Based on the assumption of 1000 days of video recording, 12 hours a day, for 10 cameras at 24 frames a second, we are currently estimating about 150 Tb of raw video data, based on samples using the current compression.

For processed data, we plan to record, for each detected fish in each frame, its ID, the camera, site, clip and frame number, where in the image it was detected, the hypothesised species name, and about 1000 descriptions augmented with a version number and certainty. We have designed a database split between the information needed for queries (about 32 bytes), and the descriptive information needed for species and behaviour recognition (about 10K bytes). Based on a current estimate of 10^{10} frames and the current average of about 2 detectable fish per frame, this suggests about 6×10^{11} bytes (600 Gb) for the query data and about 2×10^{14} bytes (200 Tb) for the descriptive data. We are currently assuming distribution across 48 machines, which means about 13 Gb/machine query data and 4 Tb/machine descriptive data.

We have then analysed how quickly it is possible to answer the '20 Questions' presented in Deliverable D2.1. From this analysis, we concluded that we should precompute by hour and by day summary files, containing a summary of the form $\{$(speciesID, $\geq certainty$,RecogAlgVersion,count)$\}$ over all species. At the most compact, the day summaries reduce to about 120 Mb, which we assume can be incore on each machine.

Using this data distribution, we analysed 'typical' queries, such as **How many different types of Species ID and their quantity T appear in each clip K in this range of dates D-D with certainty $\geq C$ for site S?** By distributing the 10^4 day summary files randomly across all machines (to encourage load balancing), we estimate that this sort of query can be estimated on the full 1000 day dataset in about 2 seconds. With suitable precomputation of summaries, we hypothesise that most of the 20 queries and those of similar structure can be computed in a few seconds.

This analysis will need to be repeated after the details have settled on the processor, incore memory and disk architecture have stabilised. Most effective is the pre-calculation of various summary records, which can be done on a nightly basis when data is no longer being collected.

2.4.5 T4.5 Support code parallelisation

No action on this task during year 1.

2.5 WP 5: System Integration and Evaluation

The main idea is that the different software modules developed (by the different partners) in the system communicated by means of the storage facility(s). This means that the data that is processed by the components is available to all partners in the project, but more importantly to the end-user. This allows the different partners to create their own components with the only dependency the access to the database. There is a database that will collect and store the data, but also allows us to query and retrieve the data again.

There will be a lot of data, so we will have to use a distributed storage facility in the future. This data also comes in different formats, like video, image, ontologies, for which we can use different sort of databases (SQL, triple stores), that can deal with the different formats. However, by using the database component, we intend to give the computer vision components a simple interface to the storage facilities without having to worry for instance about storing information in a distributed manner or different interfaces to retrieve different kind of information.

2.5.1 T5.1 - Define component interfaces

For the Fish4Knowledge project, each of the partners has to create their own component(s). The exact implementation of the component is up to the partners, however the information flow (database definition) between the partners has been defined in order to cooperate with each other. Each of the components has an input and an output, which will be defined for each component separately. To make a distinction between the importance of certain inputs and outputs, we labelled the outputs as: minor, feature, suggested, and necessary. The focus in the first stage should be on the necessary and suggested outputs. If those are finished other outputs can be added as new features. The users of the system (marine biologists) can also ask for different features.

Because the Fish4Knowledge project is under development, we expect many changes in the different components. Because we do not want to lose information, every component gets a unique identifier (by means of a lookup table, purpose and version can be retrieved). The component has to use this identifier while storing information. This means that we can also track information from old components and versions. We can easily add new components for the detection and recognition, or add newer improved version of the component. At the moment, we have a version management server (using GIT) running allowing users to check in their code and to share the code with the other partners.

In Figure 8, we show a UML diagram of all the components. We also give a short description of the input and the output of these components. More details are given in Deliverable 5.1. Notice however that all components connect to the database, allowing them to store their information and retrieve information from other components.

The exact Datastore Definition in the databases will be maintained by CWI. The Datastore Definition can be found in Deliverable 5.2 and will be extended over time if more definitions are necessary in the project. Because all the components use the database to communicate, it

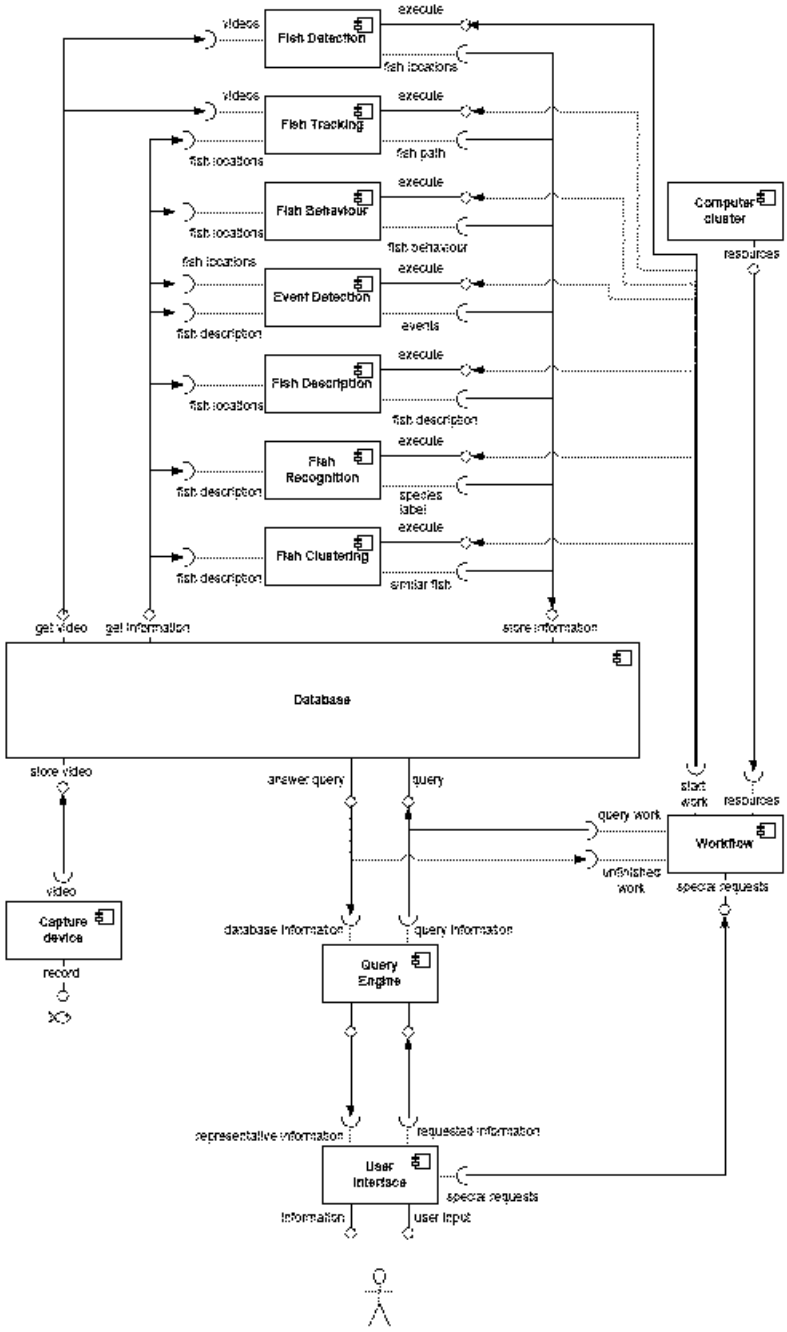


Figure 8: UML Component Diagram, showing the input and output relations of the different components

is important that the database definitions do not change often and that there is agreement about these definitions. For the computer vision, at the moment we use a relational database scheme. In addition, an RDF schema has been defined in order to expose the project data in a Linked Data-compliant solution for Web-scale sharing of resources and experimental data.

2.5.2 T5.2 - Integration and evaluation planning

To integrate the components, it is important to solve the dependencies between components at an early stage. In our case, most components have three dependencies: database access, access to videos and information of other components.

Database Access: Each program needs input from one or multiple partners in the project. The first thing that needs to be developed is a (temporary) database component (at the moment located in UCATANIA). The database component provides access to all the data in the project. In the beginning of the project, we assume that a simple SQL database and interface to this database will be sufficient for all the partners. Better solutions (distributed databases, triple stores, etc) will be developed during the project, once we have more information to make better decisions on these issues.

Fast methods to retrieve video stream/frames: There is an interface to access the video recordings, where both user/components can immediately access the videos. The video recordings are not stored in the database, but they still are used by almost all partners in the project. At the moment, we have a webinterface in Taiwan which allow the partners to retrieve the videos, this interface might be improved in the future to allow also frame retrieval.

Information about other components: Although every component should be able to access the database, it is necessary to have the same definitions and have clear definitions. In order to start, UCATANIA and UEDIN made a first database design with data that they expect to create. CWI has been involved in this design and will help to improve this based on the user requirements. Once we have this first database design, CWI will manage/maintain the datastore definitions. This means that adding, changing and removing definitions must be discussed with CWI.

NARL is responsible for the computers and the platform where all the software is running. They are looking into virtual machines to offer the partner in the project a virtual environment that allows them to create and test their components. NARL already provided the partners access to their servers. They are also going to give some manuals describing how to run jobs distributed. The partners are responsible to test their components on the system provided by NARL. NARL is responsible to support the partners and to find alternative solutions if the platform is not able to run programs required by the partners.

The performance of the entire system and the separate components is very important. In order to monitor this performance, we need to be able to evaluate the system and the separate components. The entire system consists of different components. These components have to be evaluated in different ways. The creators of the components probably know the best manner to evaluate their components. In order to contribute in their scientific fields, they have to evaluate

their components anyway. For this reason, an evaluation plan should be finished in 17 month of the project. These separate evaluation plans of each component are used as input to construct the evaluation plan of the entire system. The evaluation plan will be put on the wiki pages of the project. Based on the user's comments, new goals are set for the different components allowing everybody to improve their components on specific issues that occurred during testing. An example of evaluation which has already started is the performance of the computer vision components. In order to evaluate those components, fish images are annotated for both the fish detection and fish recognition, which is in some cases a challenge. This allows us to evaluate the quality of the fish detection and recognition component.

2.5.3 T5.3 - First integration and evaluation phase

No action on this task during period 1.

2.5.4 T5.4 - Second refinement and evaluation phase

No action on this task during period 1.

2.5.5 T5.3 - First integration and evaluation phase

No action on this task during period 1.

2.5.6 T5.4 - Second refinement and evaluation phase

No action on this task during period 1.

2.6 WP 6: Project Dissemination

This section describes our progress so far and future plans regarding project dissemination work as described in WP6.

2.6.1 T6.1 - Project Web Site Development and Availability

As part of the F4K project a publicly available project web site has been set up and running since the start of the project. Its URL is: `www.Fish4Knowledge.eu`.

The web site gives an overview of our project work and objectives and what we intend to do to achieve our objectives. It also lists personnel that are involved in this project as well as project generated resources and news releases.

The project web site is highly visible via the Google search engine. When searching the web using the keyword "Fish4Knowledge" (the project short name, it comes up in the first place. The web site will continue to be enriched and populated with our new project results, news release and resources over the lifetime of the project.

2.6.2 T6.2 - Scientific workshops

As one of the three proposed scientific workshops in our project plan, so far we have held one invited session entitled "Intelligent Workflow, Cloud Computing and Systems" as a part of the KES AMSTA conference, during June 29 - July 1 in 2011. KES AMSTA (International Conference on Agents and Multi-agent Systems - Technologies and Applications) is an international scientific conference for research publishing in the field of agent and multi-agent systems. Its interests also include knowledge representation and systems, semantics, ontologies, computational complexity, intelligent workflow and cloud computing and systems. This is an interesting and highly relevant conference for us. The invited session allowed us to create new interests in our work. It also provided a communication platform to discuss issues in more depth. In total, there were 5 talks presented at the invited session and about 20 people attended the session. Out of these 5 talks, four papers were published as a part of the main conference. One of the papers was from a F4K researcher.

This special session generated good interest and the conference organisation has identified our topic as new and interesting. We are invited back to run a similar session next year. In addition, we are invited to create and run a special edition for the International Journal of Knowledge Based and Intelligent Engineering Systems (`www.kesinternational.org/journal/`). The conference chair of KES-AMSTA, Prof. Jim O'Shea, Manchester Metro University, is due to visit and give a talk in Edinburgh on Nov 24, 2011. We are planning for an exchange visit with his group, as appropriate.

As for the next stages, in terms of running scientific workshops, we have the following plans:

- To run a follow-on invited session on the topic of "Intelligent workflow, cloud computing and systems", as a part of the KES AMSTA conference, in June 25-27, 2012. The preparation of this invited session is now in place.
- To run a workshop on the topic of "Visual observation and analysis of animal and insect behaviour", in November 2012.
- To run a workshop on the topic areas of interfaces for ground truth labelling, dates to be decided.
- To run a workshop on the topic areas of high performance computing applications for image and video analysis and processing, dates to be decided.
- To run a special session for the International Conference on Image Processing on image and video analysis under extreme real-life conditions in 2012.

In addition, we have the following plans regarding research publications:

- A special journal issue on intelligent workflow and cloud computing and systems for the International Journal of Knowledge Based and Intelligent Engineering Systems. We plan to organise this towards the end of 2012.
- A book proposal for academic publishing to cover the overall F4K project work and other related work. We plan to organise this towards the end of 2012 when we have more project results to work with.

2.6.3 T6.3 - Two Web-Mounted User Interfaces

We have developed and published a web-mounted user interface in Second Life, The Fish4Knowledge Second Life Gallery and Under-Water Aquarium, which is part-sponsored by the University of Edinburgh.

Second Life is a three-dimensional virtual world platform that is open for user developments. It also provides free memberships for everyone to use their applications. Everyone is allowed to create one "avatar" for free, so that one can roam around the virtual world to visit new sites and explore new things, and communicate with other people live or off-line. Second Life is well-known for its communicational and educational values. It is also particularly attractive, because it adds the "fun life-like" factor that a normal two-dimensional user interface could not offer.

Currently, the F4K project has set up a gallery and an under-water aquarium that is built in two levels, the F4K SL building. The ground level displays our academic research results, whereas the underground level exhibits an interactive virtual world aquarium; where virtual fish and sea life are animated and the real facts about them are on display. In addition, we are able to display and stream videos captured in the observed Taiwanese sea waters.

Currently, we have completed our first stage of the development of the F4K SL building. We have the physical building and aquarium ready to use and have populated them with essential information and artifacts. The upper level has a set of poster display boards that tell about the project. We expect to continue to enrich and update them with additional/new project results as they become available.

Currently, we have developed a simple user query interface to communicate with the users that is the front-end of the workflow engine. The more sophisticated user-goal-aware, interaction-rich and web-based user interface is currently under investigation and will be developed at later stages of the project.

2.6.4 T6.4 - Interacting with the Marine Biology Community

In addition to interacting with marine biologists that are already on our scientific advisory panel, we have also reached out to additional potentially interested scientists and practitioners. As we continue to do so, this is still work in progress.

The UCATANIA and UEDIN teams have been in discussions with Dr Owen Day, who is Head of Communications and Biodiversity and co-director of the CARIBSAVE Partnership, over how we might cooperate with our technical capabilities and their Caribbean sealife monitoring project. At the moment, some sort of shared PhD project looks likely.

