National Institute of Standards and Technology
U.S. Department of Commerce

Federal Office for Information Security

NFIQ 2.0 Framework

Michael Schwaiger
secunet Security Networks AG

Bundeskriminalamt

Fraunhofer IGD

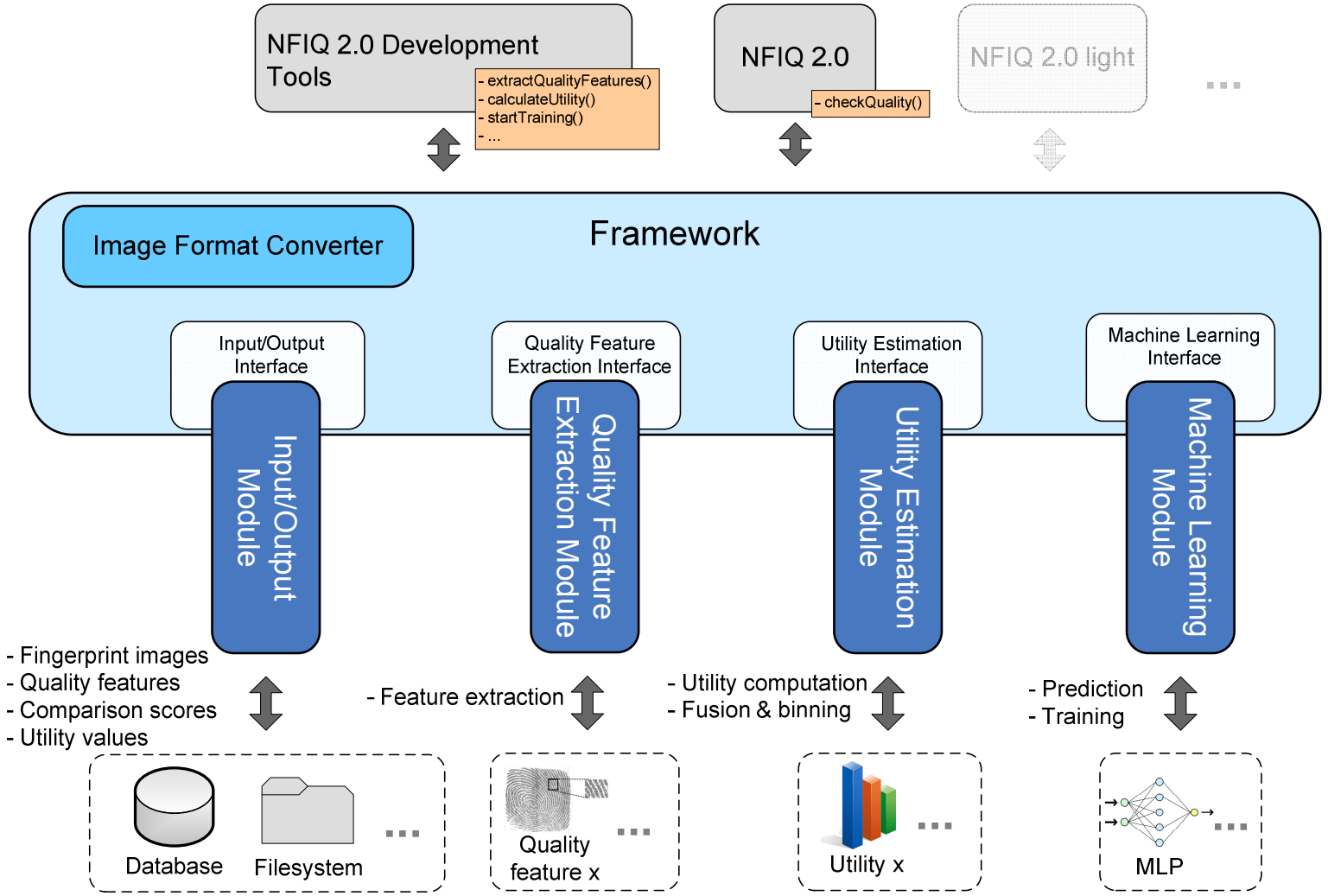
CASED

secunet

Motivation for NFIQ2.0 Framework

- Lessons learned from NFIQ re-training in 2009/2010
  - NBIS source code changes necessary for adaptation of
    - machine learning algorithm
    - feature vectors used
- Modular approach for NFIQ2.0 development is desired
  - to be flexible regarding the implementation
  - to have a common basis of functionality needed for NFIQ2.0 development which might then be extended by exchange of certain modules
  - because project team is distributed and located all over the world
  - because only certain project partners have access to certain fingerprint databases
  - because work can be shared and re-used by others
  - to simplify the development process

Architecture of NFIQ2.0 Framework



NFIQ2.0 development tools and interchange file format

- Implemented on top of the NFIQ2.0 Framework
  - ComputeQualityFeatureData
  - ComputeUtilityValues
  - StartTraining
  - ComputeQuality (final or intermediate NFIQ2.0 algorithm)
  - XMLExportImport
- XML interchange file format defined
  - exchange of training and evaluation data among project partners
    - fingerprint comparison scores of several databases
    - utility values and quality scores
    - quality features
  - referenced by unique IDs

Example: How to perform training with the NFIQ2.0 Framework I

- Assume that necessary data for training is stored in database X
- Compute quality features

```
computeQualityFeatureData(X, <featureID_1>, ..., <featureID_M>)
```

- Compute and fuse utility values

```
computeUtilityValues(X, <providerID_1>, <utilityID>)
...
computeUtilityValues(X, <providerID_N>, <utilityID>)

fuseUtilityValues(X, <providerID_1>, ..., <providerID_N>, <utility_ID>)
```

- Select images for training

```
defineImagesForTraining(X, <trainingSet_X>, <testSet_X>)
```

and/or

```
partitionDataForTraining(X, <seed>, <N_train_X>, <N_test_X>)
```

- Start training

```
train(<providerID_1>, ..., <providerID_N>, <utility_ID>, <featureID_1>, ...,
 <featureID_M>, useWeights)
```

Example: How to perform training with the NFIQ2.0 Framework II

- Select images for evaluation

  `defineImagesForEvaluation(X, <evaluationSet_X>)`

  and/or

  `partitionDataForEvaluation(X, <seed>, <N_eval_X>)`

- Start evaluation

  `evaluate(<providerID_1>, …, <providerID_N>, <utility_ID>,`
  `<featureID_1>, …, <featureID_M>)`

Federal Office
for Information Security

Contact

secunet Security Networks AG
Michael Schwaiger
michael.schwaiger@secunet.com