NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for naturvitenskap og teknologi
Institutt for kjemisk prosessteknologi

SPECIALIZATION PROJECT 2015

TKP 4550

PROJECT TITLE:

Implementing a Subsea Oil -Water Separation System in Modelica

By

Víctor Fernández Iniesta

Supervisor for the project:
Associate Professor Johannes Jäschke

Date:
18.12.2015

Abstract

The aim of the project was to obtain a Modelica model of a subsea liquid-liquid separator.

A subsea water-oil separation system consisting in one horizontal gravity separator and two swirl separators has been modeled in the equation based, object-oriented language Modelica. The units can be simulated separately or the overall system. The system can be connected in bigger systems. The modeling methodology, how the final model of the cycle is carried out, is described in the report. A new combination of separation was proposed and simulated.

A steady-state solution of the Modelica model was obtained, an alternative system was simulated.

Preface

This report is reflecting the work done as part of the specialization project (TKP4550) in the fifth year of the study program of MSc in Chemical Engineering at the Norwegian University of Science and Technology.

I would like to thank my supervisor Professor Johannes Jäschke, for their suggestions, guidance and support through the project. I would also like to thank Christoph Backi for his help regarding Modelica.

_____

Signature

_____

Date

Contents

List of figures

List of tables

1 Introduction

The use of subsea technology is nowadays increasing allowing operate reservoirs in remote places and in extreme conditions, being a cost effective alternative [1].

The goal is maximize the production and lay the ground for further growth, to achieve this it is needed to find compact and environmental friendly solutions, improve recovery from the reservoirs, permit long tie-backs to land, and developing equipment for ultradeep water.

Subsea separators have stricter limitations in comparison with topside separators, for existing facilities the space is minimal and the degree of compactness is a must.

The expected life of subsea equipment can have a huge impact on project economics. Because of high intervention costs and potential loss of production, reliability and availability of the system must be kept high.

Deepwater intervention will be costly because of the need for specialized equipment and support infrastructure. Keeping the subsea equipment light and compact will allow the use of less costly and more abundant dive-support vessels rather than having to use heavy-lift vessels.

It is needed to choose the best compromise for cost, function, operability and maintainability.
Subsea liquid-liquid separation is still a challenging issue. Technical problems have to be faced in terms of interface control, water quality, outlet water cut or solid removal.[2]

This technollogy offers some advantages as accelerates and increases production and recovery, prolongs the economic life of the fields, provides a flexible sollution for changing operating conditions and debottlenecks flowlines [3].
Due to these reasons an accurate control and optimization is key for the subsea separators.

This project was motivated by first of all reproducing parts of the work P. FürstTyvold [4] did in his Master Thesis, which was done in MATLAB, and implement a close to similar model in Modelica.

The intention of using Modelica was to avoid the causality which dominates many programming languages and take advantage of the benefits Modelica provides allowing with the use of connectors the simulation in bigger systems.

Once a model is written, the Modelica engine compiles the code into C-code or XML whereby almost anything can be further analyzed (like optimization).

Several books, articles and internet forums have been frequently employed to gain knowledge about Modelica usage, but the book by P.Fritzson [6][7] have been used as base material to build a model.

2 Process Description

2.1 Model inputs

The flow diagram for the separation system is depicted in Figure 2.1 and physical process dimensions[1] are announced in Table 2.2 empirical parameters for swirl separators are given in Table 2.1, finally the estimated optimal operation made by FürstTyvold is showed in Table 2.3

The gravity separator does a bulk separation while dewaterer and deoiler purifiy the water-rich and the oil-rich products, respectively.

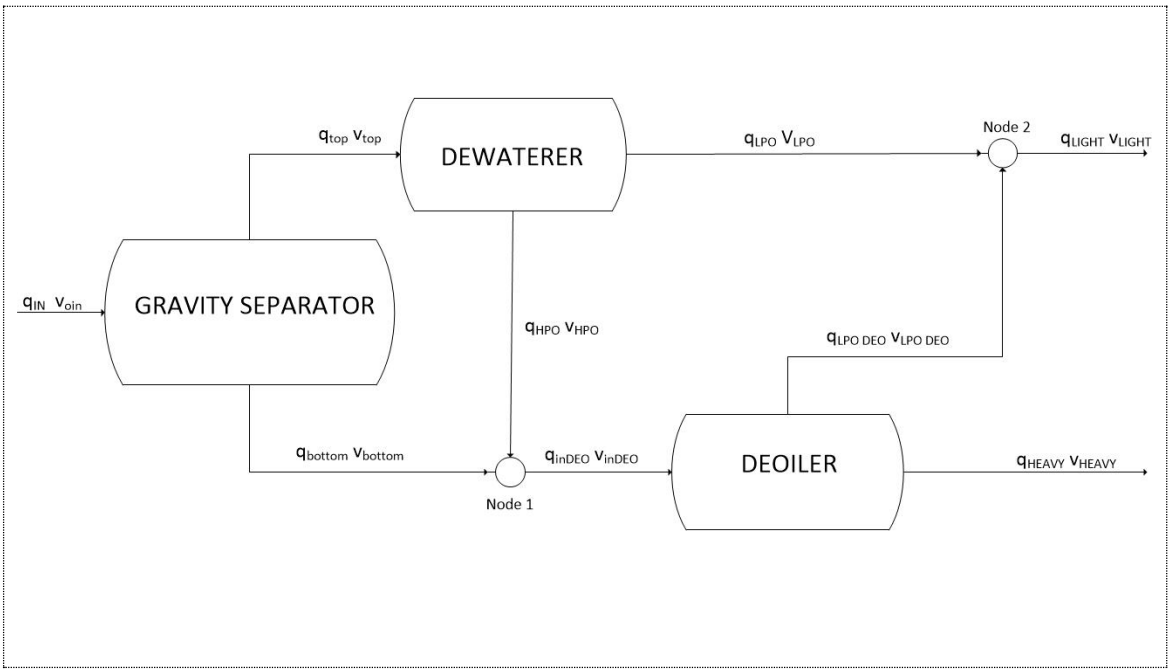

Figure 2.1: Flow diagram for the separation system. The subscript of each stream corresponds to the values in the code

[1] The variables presented are inherited from the exact same separation system defended by P.FürstTyvold's Master Thesis [4].

R_c/R [-]	0.25	-
C_{decay} [-]	0.04	-
$K_{re\text{-}en}$ [m^2]	$2 \cdot 10^{-4}$	-
$D_d(\ v^{\Theta}_{max})$ [µm]	$(-107 \cdot v^{\Theta}_{max}+600) \cdot 10\text{-}6$	$v^{\Theta}_{max} \leq 4.45$
	$(-8v^{\Theta}_{max} \cdot +160) \cdot 10\text{-}6$	$v^{\Theta}_{max} > 4.45$

Table 2.1 Empirical parameters for swirl separators

Process unit	Condition description	Symbol	Value
Gravity separator	Length	L[m]	7
	Outer pipe	R[m]	1.7
	Weir height	H_w[m]	2.55
	Droplet diameter	D_d[µm]	120
Deoiler (Swirl separator for continuous water phase)	Length	L[m]	1.7
	Outer pipe	R[m]	0.05
	Inner pipe	R_i[m]	0.025
	Swirl number large/strong/weak	Ω [-]	7.0/5.0/3.0
Dewaterer (Swirl separator for continuous oil phase)	Length	L[m]	1.7
	Outer pipe	R[m]	0.05
	Inner pipe	R_i[m]	0.043
	Swirl number large/strong/weak	Ω [-]	7.0/5.0/3.0

Table 2.2: The design conditions for the refrigeration cycle

The inputs used in oil-in water separator were based in experimental data of Van Campen[5]. No experimental data available were found for the gravity separator and water-in-oil separator. Therefore a set of dimensions estimated by P.FürstTyvold[4] were taken.

Variable	$q_{in} = 20 m^3 \cdot h^{-1}$
FS_G [-]	0.33
α_t [-]	0.94
q_t $[m^3 \cdot h^{-1}]$	6.7
α_b [-]	0.13
q_b $[m^3 \cdot h^{-1}]$	13.3
FS_{DW} [-]	0.91
$\alpha_{LPO\ DEW}$ [-]	0.98
$q_{LPO\ DEW}$ $[m^3 \cdot h^{-1}]$	6.1
$\alpha_{HPO\ DEW}$ [-]	0.5
$q_{HPO\ DEW}$ $[m^3 \cdot h^{-1}]$	0.6
$\alpha_{in\ DEO}$ [-]	0.15
$q_{IN\ DEO}$ $[m^3 \cdot h^{-1}]$	13.9
FS_{DEO} [-]	0.15
$\alpha_{LPO\ DEO}$ [-]	0.83
$q_{LPO\ DEO}$ $[m^3 \cdot h^{-1}]$	2
α_{HEAVY} [-]	0.03
q_{HEAVY} $[m^3 \cdot h^{-1}]$	11.9
α_{LIGHT} [-]	0.94
q_{LIGHT} $[m^3 \cdot h^{-1}]$	8,1

Table 2.3 Optimal operation for fixed inlet flow rate optimized by P.FürstTyvold

2.2 Fluid properties

The fluid properties were chosen following the FürstTyvold Thesis which in turn, he uses the experiments of Van Campen [5].
The water phase is brine made of tap water and sodium chloride (NaCl). The oil phase is a *refined mineral oil blended with zinc free additives* [5]. The densities and viscosities of the two phases are shown in Table 2.4.

Liquid	Density [kg · m^{-3}]	Viscosity [mPa · s]
Oil	881	8.8
Brine	1064	1.0

Table 2.4 Properties of the fluids used in the separators model

The models require the viscosity of the emulsion as a function of the oil cut. The viscosity is expressed by a third order polynomial function (eq. 2.1), which is fitted to experimentally measured values by Van Campen [5].

$$\mu = \mu_c(1 + a\varphi + b\varphi^2 + c\varphi^3) \qquad (2.1)$$

Where μ_c is the viscosity of the continuous phase, φ is the volume fraction of the dispersed phase and a, b, c are the fitted values for the coefficients in the polynomial.
The phase inversion point of this particular emulsion is measured to be at the oil volume fraction of 0.66. This means that the viscosity function must be divided into two different regions with different polynomial coefficients. One for oil-in-water emulsions ($\alpha \leq 0.66$) and one for water-in-oil emulsions ($\alpha > 0.66$).

The polynomial coefficients for the two viscosity equations are listed in Table 2.5.

Emulsion	Range in α [-]	a [-]	b [-]	c [-]
Oil in water	0 - 0.66	110	-400	470
Water in oil	0.66 - 1	-1.6	27	23

Table 2.5 Fitted values for the coefficients in the polynomial for the viscosity of the emulsion

3 Computer tools (Dymola)

The modeling and simulations are performed in the environment Dymola
by use of the GCC compiler. The user friendly environment is suited for both modeling, simulation and plotting of results. The GCC compiler handles most of the features present in the Modelica language.

They were defined units of each magnitude as well as the inputs as parameters to facilitate handling in the simulation environment without having to enter in the code, and making easy the understanding of every variable.

Two codes were developed, one for the simulation of each block individually and other by the use of connectors allowing simulate the complete separation system.

Due to the use of connectors the system can be implemented easily in a bigger system simply connecting the inputs and outputs of the system.

4 Modeling methodology

The system in Figure 2.1 is considered when discussing the modeling approach below. The Modelica model code is attached in Appendix A.

This model is based in the MATLAB code by FürstTyvold [4] with small variations in the model, but using the advantages of Modelica and simulating the complete system at the same time.

The initial values of the system model are the flow rate in the gravity separator, inlet oil volume fraction, flow split of each separator and swirl separators swirl number.
The outputs are the heavy outlet flow rate, heavy phase volume oil fraction, light outlet flow rate and light volume oil fraction.

4.1 Gravity separator

The model described below estimate the oil volume fractions in the outlets (Set of equations 4.1) given the inlet composition, the flow rate and the flow split of the gravity separator. Also the physical dimensions of the separator and the properties of the emulsion must be known.

The model is based on two independent plug flows flowing through the separator and exiting through two respective outlets. The oil droplets are pushed upwards by gravitational buoyancy forces and passes from the lower to the upper plug flow while an equal amount of water moves in the opposite direction. The vertical movement of the droplets were estimated assuming uniform droplet size distribution.

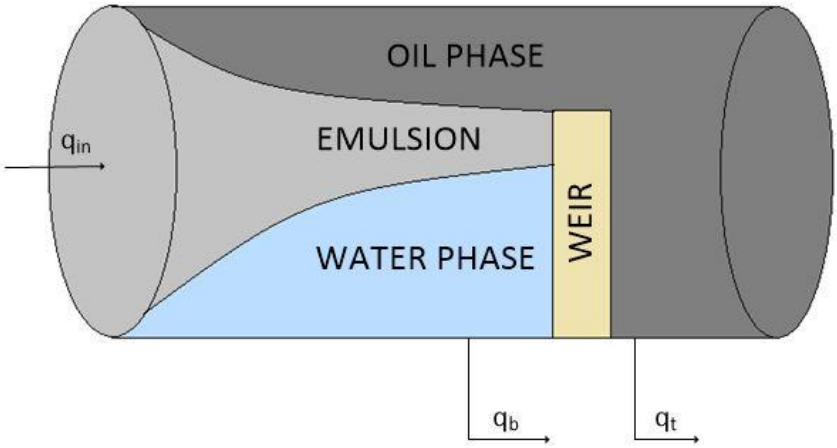

Figure 4.1 Horizontal gravity separator.

$$v_h = \frac{q_b}{A_b} \tag{4.1a}$$

$$A_b = \frac{R^2}{2} \cdot \left[2\cos^{-1}\left(\frac{R-H_w}{R}\right) - \sin\left(2\cos^{-1}\left(\frac{R-H_w}{R}\right)\right)\right] \tag{4.1b}$$

$$v_v = \frac{2r_d^2(\rho_d-\rho)g}{9\mu(\alpha)} \tag{4.1c}$$

$$\Delta h = \frac{v_v}{v_h} L \tag{4.1d}$$

$$A_e = \frac{R^2}{2} \cdot \left[2\cos^{-1}\left(\frac{R-d}{R}\right) - \sin\left(2\cos^{-1}\left(\frac{R-d}{R}\right)\right)\right] \tag{4.1e}$$

$$\alpha_b = \alpha_{in}\frac{A_e}{A_b} \tag{4.1f}$$

$$\alpha_t = \frac{1}{q_t}[\alpha_{in}q_{in} - \alpha_b q_b] \tag{4.1g}$$

Where:
v_h = Horizontal velocity of a drop moving under the weir
H_w = Weir height
q_b = Volumetric flow rate of the bottom outlet stream
A_b = Cross section area of the lower part of the separator
R = Radius of the separator
v_v = Vertical velocities of the droplets
r_d = Radius of the droplet
ρ_d = Density of the droplet
ρ = Density of the continuous phase
g = Gravitational acceleration

$\mu(\alpha)$ = Viscosity of the emulsion function of the oil cut (α)
Δh = Vertical distance of a droplet entering the separator at the bottom of the tank travel during the residence time of the separator
L = Horizontal distance from the inlet to the weir
A_e = Area of circular segment that determines the amount of oil left in the bottom part of the tank at the end of the separator
d = Segment who determines the smallest distance of the circular area A_e
α_b = Oil volume fraction of the bottom outlet
α_{in} = Oil volume fraction of the inlet
α_t = Oil volume fraction in the top outlet
q_t = Flow rate in the top outlet
q_{in} = Flow rate in the inlet
q_b = Flow rate in the bottom outlet

4.2 Deoiler

The model described below (Set of equations 4.2) provides the necessary equations to estimate the oil volume fraction in the outlet (heavy phase and light phase) given the inlet composition, flow rate and the flow split of the separator.
We need to know previously the physical dimensions of the separator, including the swirl number furthermore the model contains three parameters that must be determined (C_{decay}, $k_{re\text{-}en}$ and R_c).

Other inputs required are the physical properties of the emulsion, the densities of both phases and the correlation between the viscosity and the oil volume fraction.
The model is based on the average droplet size, providing an estimation of separation performance. The break-up of droplets will increase the swirl flow and a correlation between average droplet size and maximum tangential velocity is also required.

The density difference between the oil droplets and the continuous phase forces toward the centre of the separator the light droplets.

Re-entrainment flow rate was included in the model to compensate the simplification of the velocity profiles, to correct the existing net flux crossing the boundary between regions, this flow rate assumes that the amount of liquid re-entrained will increase when the difference in the axial velocities increase.

The centrifugal acceleration is the driving force of the separation and can be described as a Rankine vortex. A Rankine vortex has two zones, the inner region with a solid body rotation and an outer region that was simplified assuming constant velocity in this region. The separation radio between regions is at R_c. For the dewaterer separator considered in the experiment of Van Campen [5], R_i is greater than R_c , only constant tangential velocity is used in the model from (r_{in} >R_i) to R_i.

The swirling flow is expected to lose momentum caused by stress from the pipe wall, to count with it was introduced a parameter C_{decay} that can be found experimentally.

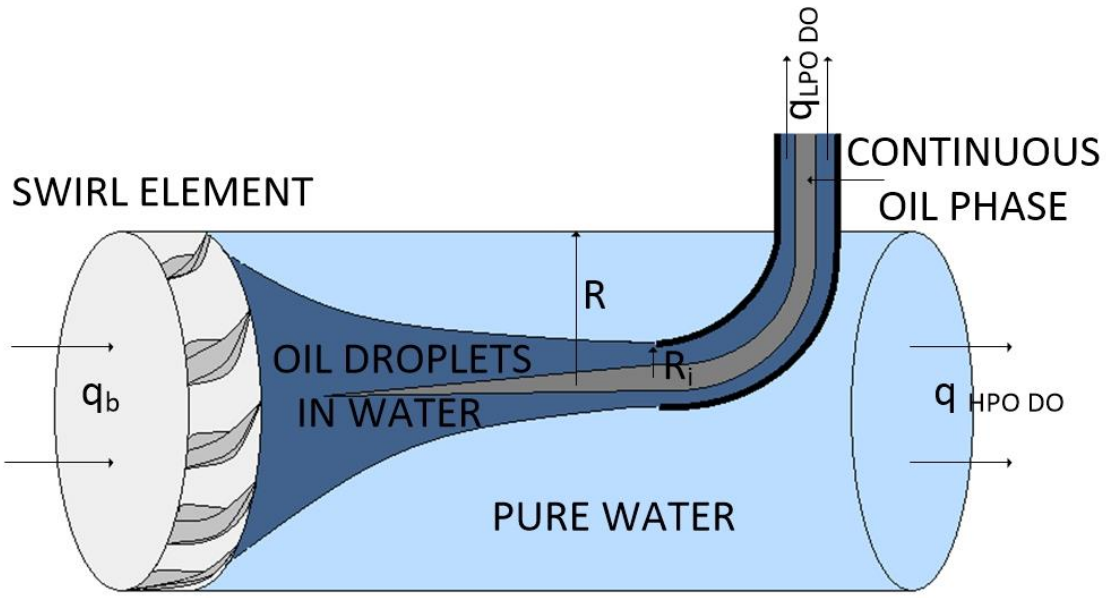

Figure 4.2 Deoiler. The swirl element introduces a rotation in the liquid. The inlet flow is separated by centrifugal buoyancy forces

$$FS = \frac{q_{LPO}}{q_{in}} \tag{4.2a}$$

$$v_z(r) = \begin{cases} \frac{q_{LPO}}{\pi R_i^2}, & 0 \le r \le R_i \\ \frac{q_{hPO}}{\pi(R^2 - R_i^2)}, & R_i < r \le R \end{cases} \tag{4.2b}$$

$$a_c(r) = \frac{v_\theta(r)^2}{r} \tag{4.2c}$$

$$v_\theta^0(r) = \begin{cases} v_\theta^{\max \frac{r}{R_c}}, & 0 \le r \le R_c \\ v_\theta^{max}, & R_c < r \le R \end{cases} \quad (4.2d)$$

$$v_\theta^{max} = \Omega v_{z,b} \quad (4.2e)$$

$$v_\theta(r,z) = v_\theta^0(r) e^{\frac{-C_{decay,z}}{2R}} \quad (4.2f)$$

$$v_r(r,z) = \frac{2r_d^2(\rho_d-\rho)v_\theta^2(r,z)}{9\mu(r,z) \quad r} \quad (4.2g)$$

$$\alpha_c(r,z) = \alpha_{in} \frac{FS(R^2-R_i^2)+(1-FS)(r_{in}^2-R_i^2)}{(1-FS)(r^2-R_i^2)+FS(R^2-R_i^2)} \quad (4.2h)$$

$$\tau = \frac{\pi(R^2-R_i^2)L}{(1-FS)q_{in}} \quad (4.2i)$$

The goal for the model is to find the radial inlet position (r) of the droplet that exits the separator at r = R_i and it is found by integrating the radial velocity (4.2g) from t = 0 to t = τ using the second order explicit Runge-Kutta integrator dividing r into 10 equal steps.

$$k_1 = v_r(t_n, r_n)\Delta t \quad (4.2j)$$

$$k_2 = v_r(t_n + \Delta t, r_n + k_1)\Delta t \quad (4.2k)$$

$$r_{n+1} = r_n + \frac{k_1+k_2}{2} \quad (4.2l)$$

Viscosity and radial velocity (4.2h and 4.2g), are function of the radial inlet position of the droplet and the radial outlet position is fixed, the governing differential equation is a boundary value problem. The inlet position that lead to the droplet exiting at (r = R_i) is founded using the shooting method (Newton-Raphson method) to solve the value boundary problem.

Assuming that the distribution of oil droplets in the radial and tangential direction is uniform at the inlet:

$$\alpha'_{LPO} = \min\left[1, \alpha_{in} \frac{FS(R^2-R_i^2)+(1-FS)(r_{in}^2-R_i^2)}{FS(R^2-R_i^2]}\right] \quad (4.2m)$$

$$\alpha'_{HPO} = \frac{\alpha_{in}-\alpha'_{LPO}FS}{1-FS} \quad (4.2n)$$

$$q_{re-en} = k_{re-en}\Delta v \quad (4.2o)$$

$$\alpha_{LPO} = \frac{1}{q_{LPO}}[\alpha'_{LPO}(q_{LPO} - q_{re-en}) + \alpha'_{HPO}q_{re-en}] \quad (4.2p)$$

$$\alpha_{HPO} = \frac{\alpha_{in} - \alpha_{LPO}FS}{1-FS} \quad (4.2q)$$

Where:

FS = Flow Split

q_{in} = Volumetric flow rate at inlet

q_{LPO} = Volumetric flow rate at light phase outlet

q_{HPO} = Volumetric flow rate at heavy phase outlet

R_i = Radius of the light phase outlet

R = Radius of the deoiler

a_c = Centrifugal acceleration

v_θ = Tangential velocity of the fluid

v^0_θ = Tangential velocity right downstream of the swirl element

R_c = Rankine radius

v^{max}_θ = Maximum tangential velocity

$v_{z,b}$ = Bulk velocity in the axial direction

Ω = Proporcionality constant, swirl number

C_{decay} = Experimental decaying factor

v_r = Radial velocity in Stokes regime

r_d = Droplet radius

μ = Viscosity of the emulsion

ρ_d = Dispersed phase density

ρ = Continuous phase density

τ = Residence time of the droplet

L = Distance between the swirl element and the inner pipe

r_n = Radial position of the droplet

Δ = Runge-Kutta time step

V_r = Radial velocity of the droplet

α'_{LPO} = Oil volume fraction in the LPO assuming distribution of oil droplets in the radial and tangential direction is uniform at inlet

α'_{HPO} = Oil volume fraction in the HPO assuming distribution of oil droplets in the radial and tangential direction is uniform at inlet

α'_{in} = Oil volume fraction in the inlet assuming distribution of oil droplets in the radial and tangential direction is uniform at inlet

R_i = Radius of the inner pipe

r_{in} = Inlet position of the droplet that exits at $r = R_i$

q_{re-en} = Re-entrainment rate

k_{re-en} = Proportionality constant of re-entrainment

Δv = Velocity difference between the inner and the outer region

α_{LPO} = Final oil volume fraction in the LPO

α_{HPO} = Final oil volume fraction in the HPO

4.3 Dewaterer

The model described below (Set of equations 4.3) predicts the oil volume fraction in the outlet streams govern the inlet composition, flow rate and the flow split, for water in an oil phase. The model is based on the same physical laws as the Deoiler (Oil in water phase).
The main difference is that density of the droplets is greater than the density of the continuous phase being forced the droplets by the centrifugal buoyancy forces outwards.

The equations are the same than 4.2 set except for the equation 4.2d in Deoiler the tangential velocity didn't operate changing in the discontinuous region but for Dewaterer are expected that droplets travel from the inner region to the outer. To avoid problems with the numerical solvers it is employed the smoothing approximation of Balakrishna and Biegler resulting in the next equation.

$$v_\theta^0 = v_\theta^{max} - max\left(v_\theta^{max}\left[1 - \frac{r}{R_c}\right], 0\right) \tag{4.3a}$$

Where:
v_θ = Tangential velocity of the fluid
$v^0{}_\theta$ = Tangential velocity right downstream of the swirl element
$v^{max}{}_\theta$ = Maximum tangential velocity
R_c = Rankine radius

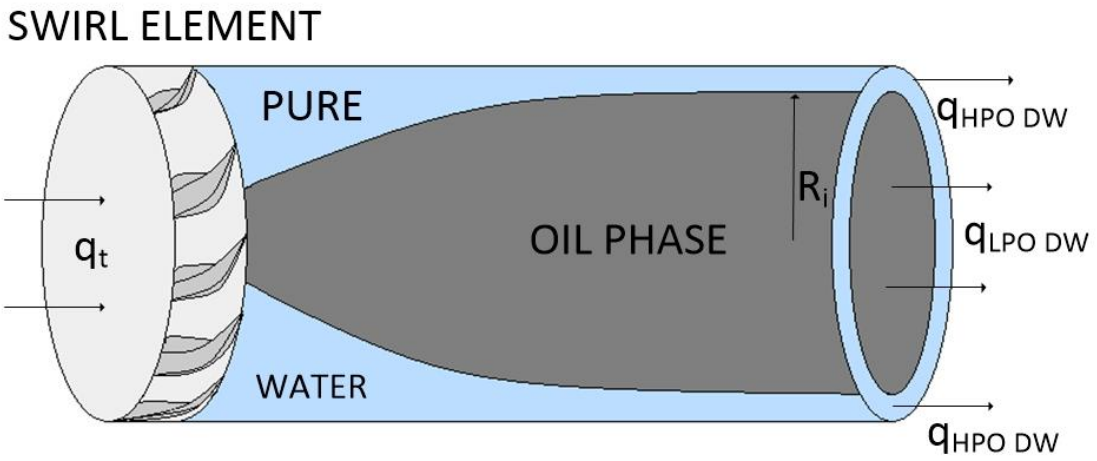

Figure 4.3 Dewaterer. The water droplets are pushed outwards by the centrifugal buoyancy forces

4.4 Nodes

Two nodes were used as mixers with two inputs and one output for the Deoiler inlet and for the light phase system outlet (Rich in oil).
Each node consists in the mass balance equations.
Node A:

$$q_{in\ DO} = q_{HPO\ DW} + q_b \tag{4.4a}$$

$$\alpha_{in\ DO} = \frac{q_{HPO\ DW}\alpha_{HPO\ DW} + q_b\alpha_b}{q_{in\ DO}} \tag{4.4b}$$

Node B:

$$q_{LIGHT} = q_{LPO\ DW} + q_{LPO\ DO} \tag{4.4c}$$

$$\alpha_{LIGHT} = \frac{q_{LPO\ DW}\alpha_{LPO\ DW} + q_{LPO\ DO}\alpha_{LPO\ DO}}{q_{LIGHT}} \tag{4.4d}$$

Where:
$q_{IN\ DO}$ = Inlet flow rate in Deoiler
$q_{HPO\ DW}$ = Outlet flow rate in Dewaterer
q_b = Outlet flow rate of the bottom in the gravity separator
q_{LIGHT} = Outlet flow rate of the light outlet in the system
$q_{LPO\ DW}$ = Outlet flow rate of the LPO in Dewaterer
$q_{LPO\ DO}$ = Outlet flow rate of the LPO in Deoiler
$\alpha_{IN\ DO}$ = Oil volume fraction in Deoiler
$\alpha_{HPO\ DW}$ = Oil volume fraction in Dewaterer
α_b = Oil volume fraction of the bottom in the gravity separator
α_{LIGHT} = Oil volume fraction of the light outlet in the system
$\alpha_{LPO\ DW}$ = Oil volume fraction of the LPO in Dewaterer
$\alpha_{LPO\ DO}$ = Oil volume fraction of the LPO in Deoiler

5 Results

5.1 The Modelica model construction

The most time spent was in making the model from scratch. We can simulate the model block by block running the model of each block individually the parameters and inputs can be modified in the simulation environment easily.

Otherwise the connected code can be simulated, the main difference is that we can modify only the inlet flow rate in the system, inlet oil cut and flow splits of each separator while the rest parameters are determined by them.

The gravity separator model was based by equations being a non-causal model however, Deoiler and Dewaterer were modeled by the use of functions.

The main function in Deoiler is swirl_func3 who determine the outlet streams of Deoiler. This function call at the solver RK2 and, RK2 call at swirl_sep2 giving the radial velocity of a droplet afterwards Deoiler function is called estimating the outlet composition having as input the greatest inlet radial position of all the droplets that exits in LPO, the shooting function also solve the boundary problem.

Dewaterer block is modeled as the same way than Deoiler.

The connectors link the blocks sharing the volume flow rate and oil volume fraction between outlets and inlets in each block, in addition they allow connect the complete system with others models facilitating the simulations with bigger system or change the system link configuration. They were defined as:

```
connector Flow_port

  import Oil_Water_Separation_System;
    import SI = Oil_Water_Separation_System.Units;
 SI.OilVolumeFraction volfracc "Volume fraction in the connection point";
 SI.VolumeFlowRate volflow "Volume flow rate in the connection point";

end Flow_port;
```

The complete system simulation code is shown below, the complete code can be found in the Appendix, initial values are the inlets found as optimal.

```
model SimulationOilWaterSystem

 Oil_Water_Separation_System.BLOCK_GRAVITY_SEPARATOR.Block_GravitySeparator GS;
  Oil_Water_Separation_System.BLOCK_DEWATERER.Block_Dewaterer DEW;
  Oil_Water_Separation_System.Node_2inlets_1outlet NODE1;
  Oil_Water_Separation_System.BLOCK_DEOILER.Block_Deoiler DEO;
  Oil_Water_Separation_System.Node_2inlets_1outlet NODE2;

equation
  connect(GS.flowporttop,DEW.flowportindew);
  connect(DEW.flowportoutdewH,NODE1.flowportinleta);
  connect(GS.flowportbottom,NODE1.flowportinletb);
  connect(NODE1.flowportoutlet,DEO.flowportindeo);
  connect(DEO.flowportoutdeoL,NODE2.flowportinleta);
  connect(DEW.flowportoutdewL,NODE2.flowportinletb);

end SimulationOilWaterSystem;
```

The code for the system consisting in gravity separator and Dewaterer with recycle of the heavy phase outlet is showed below, explanation of the system in section 5.2.5.

```
model SimulationgravityDewaterer
  Oil_Water_Separation_System.BLOCK_GRAVITY_SEPARATOR.InputStream IN(qin=0.005
5,vo_in=0.4);
  Oil_Water_Separation_System.BLOCK_GRAVITY_SEPARATOR.Block_GravitySeparator G
S(FS=0.49);
  Oil_Water_Separation_System.BLOCK_DEWATERER.Block_Dewaterer DEW(FS=0.85);
  Oil_Water_Separation_System.Node_2inlets_1outlet NODE1;

equation
  connect(IN.flowport,NODE1.flowportinleta);
  connect(DEW.flowportoutdewH,NODE1.flowportinletb);
  connect(NODE1.flowportoutlet,GS.flowportinlet);
  connect(GS.flowporttop,DEW.flowportindew);

end SimulationgravityDewaterer;
```

5.2 Validation of the results

5.2.1 Gravity separator

The model was made as an exact copy of the model made by P. FürstTyvold. The results showed of the gravity separator in the thesis differ from the MATLAB code thus, this difference was the same in the Modelica model. It is showed in the figure 5.1. It affects the inlet conditions of the swirl separators.

Note that is not expected that the model predicts accurate values for purities of the outputs in the gravity separator.

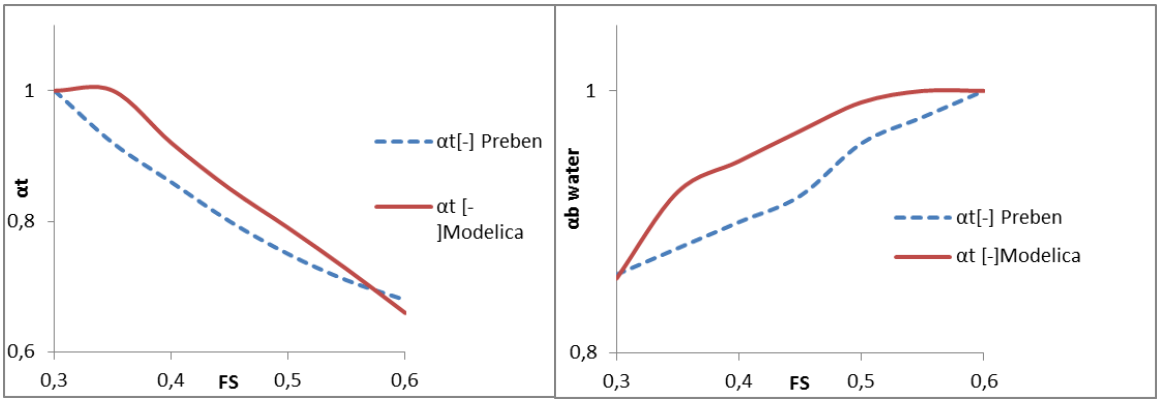

Figure 5.1 Purity of product streams versus flow split of the gravity separator. q_{in}=20m^3/h α_{in}=0.4

It is expected that the separation efficiency decreases with increases in the flow rate. The gravitational acceleration is constant in this separator.

The simulation (Figure 5.1) shows that increasing the flow split the purity of oil in the top stream decreases while the purity of water in the bottom outlet increases.

5.2.2 Swirl separator for Oil-in-Water Emulsions (Deoiler)

In order to facilitate the study of the swirl separator was defined dispersed performance as:

Dispersed performance η_{dis} is a measure about how much desired liquid exits in the expected output. It is one if all the pure oil exits in LPO and all the pure water exits in HPO.

$$\eta_{dis} = 1 - \frac{(1-\alpha_{LPO})q_{LPO}+\alpha_{HPO}q_{HPO}}{q_{in}} \tag{5.1}$$

The figure 5.2 shows the effect of the inlet flow rate on the efficiency, it is analysed in the three swirl elements large is the greater swirl number and strong the lower.
The behaviour is influenced by the empirical droplet size correlation. The local minimum correspond with the maximum tangential velocity of 4.45 $m \cdot s^{-1}$. This figure shows clearly that q_{in} should be necessary control at the maximum dispersed efficiency value.

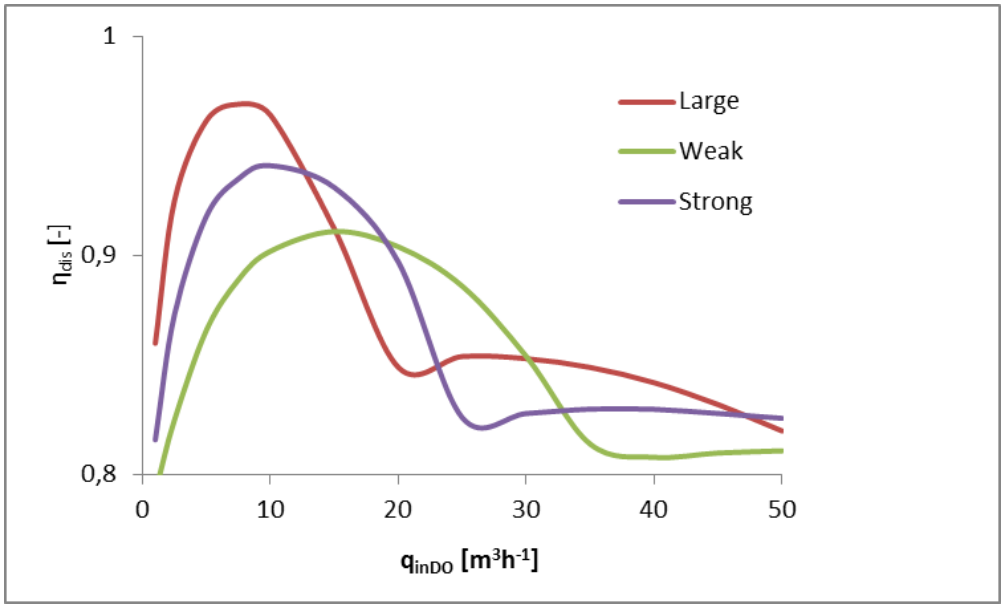

Figure 5.2 Dispersed performance versus flow rate simulated for the three swirl elements. FS = $\alpha_{in\ DO} = 0.135$

Flow split is a parameter that can be modified affecting the quality of the product streams (Figure 5.3), when the purity of the HPO increases the purity of the HPO decreases, the optimal value of the flow split depends on the main objective of the separator.
Note that the results presented by the Modelica model are same as in MATLAB model.

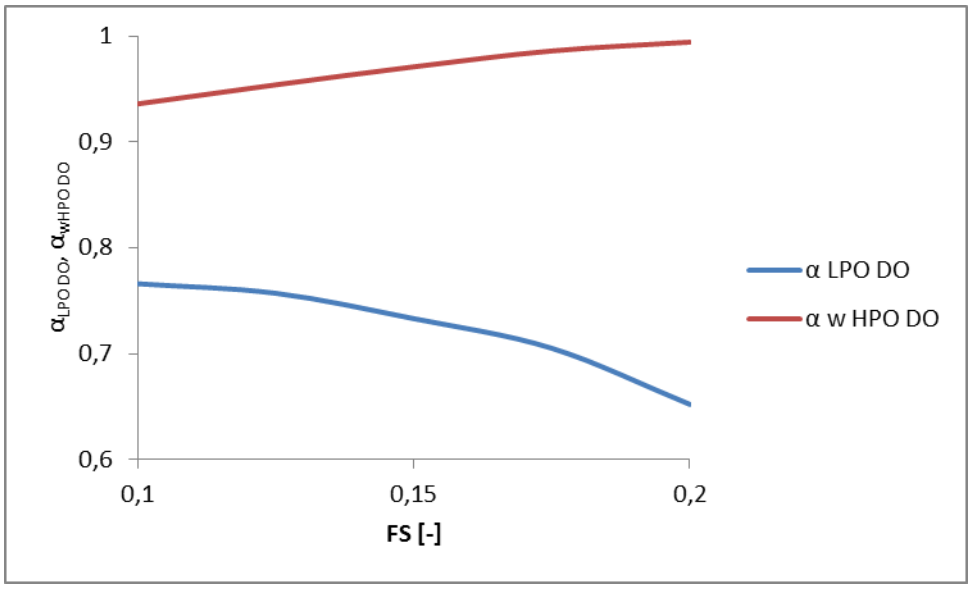

Figure 5.3 Purity of the output flow streams versus flow split inlet flow rate = 13.65m^3h^{-1} and inlet oil cut = 0.135.

5.2.3 Swirl separator for Water-in-Oil Emulsions (Dewaterer)

The performance estimated in Deoiler has the same behaviour than Dewaterer this is caused by having the same dimensions and the same droplet size equation.

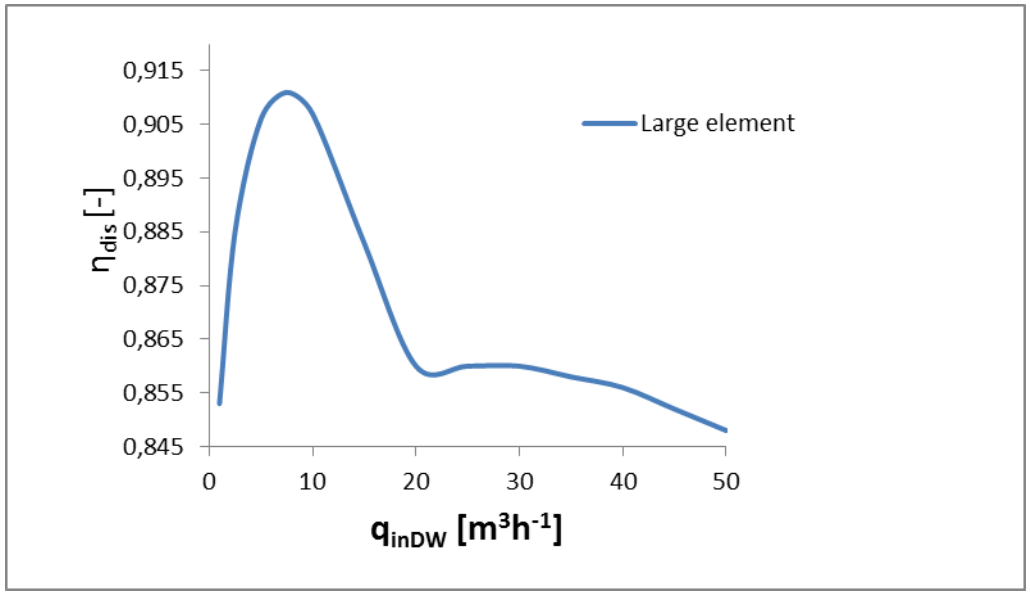

Figure 5.4 Dispersed performance versus flow rate in Dewaterer. α_{in} = FS =0.9 and long swirl element were used.

When the flow split increases, the HPO water volume fraction increases while the LPO oil volume rate decreases this is the same conclusion than in Deoiler. Depending on the purpose of the separator we can manipulate the flow split.

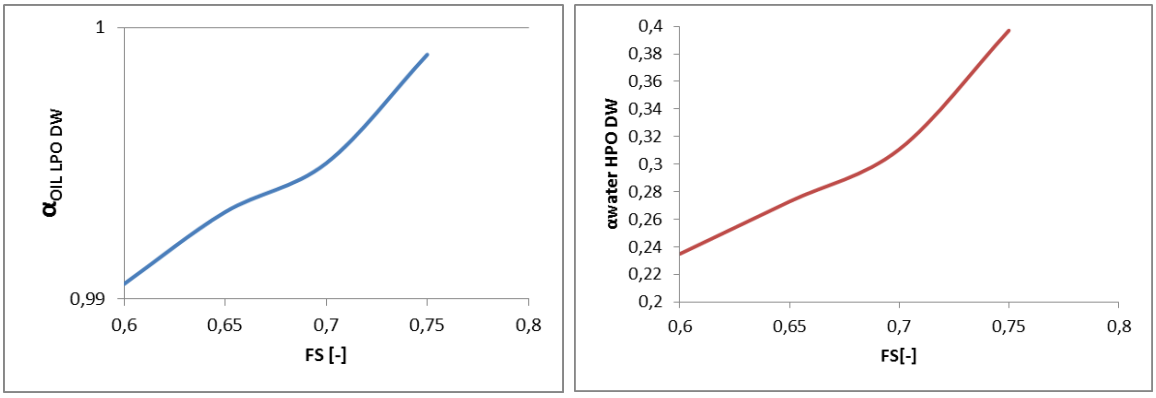

Figure 5.5 Purity of products versus flow split in Dewaterer. $q_{in\ DW} = 7.2m^3 \cdot h^{-1}$ αin =0.9 , large swirl element was used.

5.2.4 Optimal values in the system

One optimal solution was found in the system showed in Figure 2.1 manipulating flow splits with water volume fraction in outlet heavy phase $<$ 0.3 and maximizing the oil volume fraction in the light phase while the inputs were $q_{IN} = 20m^3 \cdot h^{-1}$ $\alpha_{in} = 0.4$. It was found simulating the global system. Global dispersed efficiency for this system is $\eta_{DIS\ G}$ 0.95.

Variable	$q_{in} = 20m^3 \cdot h^{-1}$
FS_G [-]	0.36
α_t [-]	0.98
q_t $[m^3 \cdot h^{-1}]$	7.2
α_b [-]	0.07
q_b $[m^3 \cdot h^{-1}]$	12.8
FS_{DW} [-]	0.88
$\alpha_{LPO\ DEW}$ [-]	0.995
$q_{LPO\ DEW}$ $[m^3 \cdot h^{-1}]$	6.33
$\alpha_{HPO\ DEW}$ [-]	0.9
$q_{HPO\ DEW}$ $[m^3 \cdot h^{-1}]$	0.87
$\alpha_{in\ DEO}$ [-]	0.135
$q_{IN\ DEO}$ $[m^3 \cdot h^{-1}]$	13.65
FS_{DEO} [-]	0.135
$\alpha_{LPO\ DEO}$ [-]	0.73
$q_{LPO\ DEO}$ $[m^3 \cdot h^{-1}]$	1.84
α_{HEAVY} [-]	0.03
q_{HEAVY} $[m^3 \cdot h^{-1}]$	11.81
α_{LIGHT} [-]	0.935
q_{LIGHT} $[m^3 \cdot h^{-1}]$	8.19

Table 5.1 Optimal values found in Modelica for complete system model were q_{IN} =$20m^3 \cdot h^{-1}$, $\alpha_{in} = 0.4$ and large swirl element.

5.2.5 New system Gravity separator and Dewaterer

A new system distribution was simulated avoiding the use of Deoiler showing good results, the difference is that the Light Phase Output lose 7% of purity, a cost-analysis should be made.

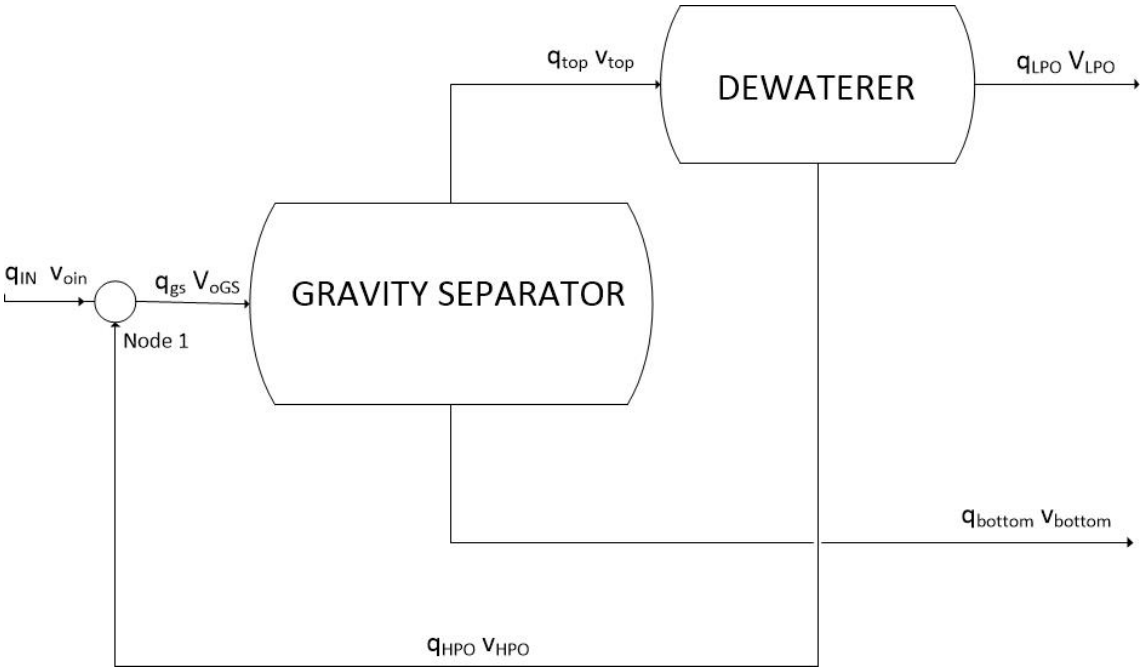

Figure 5.6 System consisting in gravity separator and deoiler with recycle of heavy phase stream.

The optimal results maintaining the water phase output with 0.03 volume fraction in oil and maximizing the oil volume fraction in light output. The $\eta_{DIS\,G}$ (dispersed efficiency) is 0.92

Variable	$q_{in} = 20m^3 \cdot h^{-1}$
FS_G [-]	0.49
$\alpha_{in\ GS}$ [-]	0.39
$q_{in\ GS}$ [$m^3 \cdot h^{-1}$]	21.57
α_t [-]	0,78
q_t [$m^3 \cdot h^{-1}$]	10,47
α_b [-]	0.03
q_b [$m^3 \cdot h^{-1}$]	11
FS_{DW} [-]	0.85
$\alpha_{LPO\ DEW}$ [-]	0.86
$q_{LPO\ DEW}$ [$m^3 \cdot h^{-1}$]	9
$\alpha_{HPO\ DEW}$ [-]	0.39
$q_{HPO\ DEW}$ [$m^3 \cdot h^{-1}$]	1.57

Table 5.2 Optimal values found in Modelica for model gravity separator and dewaterer where $q_{IN} = 20m^3 \cdot h^{-1}$, $\alpha_{in} = 0.4$ and large swirl element.

6 Discussion

The model shows the same result that the model that was based except for gravity separator discussed in (Section5.2.1).
Another proposal was simulated avoiding the use of Deoiler losing 2% of performance.

The advantage of use the Modelica model developed is the versatility using the connectors, allowing create quickly modifications as seen in 5.2.5. If it is needed reoptimization with tools as JM compiler a complete equation based model should be made.

The gravity separator model should be improved with experimental values and the behaviour of the interaction between the two phases.

Future Work:

- Developing a model 100% conciliable with Subpro library, especially attention should be paid to the physical properties of the fluid, basing in the media used for this library.
- Applying a control structure and optimization using self-optimizing control.
- Integrating the liquid-liquid system in the complete subsea module and re-optimize.

List of symbols and abbreviations

Symbol[2]	Dimension	Description
Latin letters		
a		Polinomial coefficient
A_b	m^2	Cross section area of the lower part of the gravity separator
a_c	$m \cdot s^{-2}$	Centrifugal acceleration
A_e	m^2	Area of circular segment that determines the amount of oil left in the bottom part of the tank at the end of the gravity separator
C_{decay}	-	Experimental decaying factor
d	m	Segment who determines the smallest distance of the circular area A_e
D_d	μm	Droplet diameter
FS	-	Flow split
g	$m \cdot s^{-2}$	Gravitational acceleration
H_w	m	Weir height
L		Length
R	m	Radius
$q_{re\text{-}en}$	m^2	Re-entrainment flowrate
v_h	$m \cdot s^{-1}$	Horizontal velocity of a drop moving under the weir
v_v	$m \cdot s^{-1}$	Vertical velocities of the droplets
$v_{z,}$	$m \cdot s^{-1}$	Bulk velocity in the axial direction
Greek letters		
α	-	Oil volume fraction of the bottom outlet
Δ	-	Runge-Kutta time step
η_{DIL}	-	Diluted efficiency
η_{DIS}	-	Dispersed efficiency
μ	$kg \cdot s^{-1} \cdot m^{-1}$	Viscosity

[2] *Continued on next page*

Symbol[3]	Dimension	Description
ρ	$kg \cdot m^{-3}$	Density
τ	-	Residence time of the droplet
Ω	-	Swirl number
Superscript		
‘	-	uniform distribution at inlet
o	-	Right downstream of the swirl element
max	-	Maximum
Subscript		
b	-	Bottom
d	-	Droplet
DE, DEO	-	Deoiler
DW, DEW	-	Dewaterer
GS	-	Gravity separator
HPO	-	Heavy phase outlet
i	-	Input
in	-	Inlet
LPO	-	Light phase outlet
t	-	Top
θ	-	Tagential

[3] *Continued from previous page*

References

[1] STL technologies, Internet, November 2015.
http://www.subseatek.com/

[2] SPOOLSEP- *"Subsea gravity liquid-liquid separator"* Internet, December 2015.
http://www.saipem.com/sites/SAIPEM_en_IT/scheda/spoolsep.page

[3] Maximize Recovery *"SubSea Separation"* Internet, November 2015.
http://www.maximizerecovery.com/Technologies/subsea-separation

[4] P. FürstTyvold, *Modeling and Optimization of a Subsea Oil-Water Separation System.* Master thesis, Norwegian University of Science and Technology, June 2015.

[5] L. VanCampen *Bulk Dynamics of Droplets in Liquid-Liquid Axial Cyclones. PhD thesis,* Technische Universiteit Delft, January 2014.

[6] P.Fritzson. *Introduction to Modeling and Simulation of Technical and Physical Systems with Modelica.* Ed.Wiley, 2011.

[7] P.Fritzson. *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1* Ed.Wiley, 2004.

Appendix A: Modelica Code

This appendix contains the Modelica code created during the project.

- Appendix A.1 Connector.
- Appendix A.2 Units used in the models.
- Appendix A.3 Deoiler (Water continuous phase).
- Appendix A.4 Dewaterer (Oil continuous phase).
- Appendix A.5 Gravity Separator
- Appendix A.6 Node.
- Appendix A.7 Numerical solvers.

A.1 Connector

```
connector Flow_port

  import Oil_Water_Separation_System;
  import SI = Oil_Water_Separation_System.Units;

 SI.OilVolumeFraction volfracc "Volume fraction in the connection point";
 SI.VolumeFlowRate volflow "Volume flow rate in the connection point";

end Flow_port;
```

A.2 Units used in the models

```
package Units

  type Area = Real(final quantity = "Area", final unit = "m2", min = 0);
  type DynamicViscosity = Real(final quantity = "DynamicViscosity", final unit = "Pa.s",
                               min = 0);
  type Density = Real(final quantity = "Density", final unit = "kg/m3", displayUnit =
                      "g/cm3 ", min = 0.0);
  type Length = Real(final quantity = "Length", final unit = "m");
  type Velocity = Real(final quantity = "Velocity", final unit = "m/s");
  type VolumeFlowRate = Real(final quantity = "VolumeFlowRate", final unit = "m3/s");
  type OilVolumeFraction = Real(final quantity = "OilVolumeFraction", final unit = "1", min
                                = 0, max = 1);
  type FlowSplit = Real(final quantity = "FlowSplit", final unit = "1", min = 0, max = 1);
  type Efficiency = Real(final quantity = "FlowSplit", final unit = "1", min = 0, max = 1);
  type SwirlElement = String(final quantity = "Swirl element weak, strong or large",
                             final unit = "adim");
```

```
17
18  end Units;
```

A.3 Deoiler (Water continuous phase)

A.3.1 swirl_sep2

This function contains the most part of the Deoiler model equations, returns the radial velocity of a droplet.

```
1   function swirl_sep2
2
3    input Real t;
4    input Real x;
5    input Real ini[:];
6    //Array size defined at function call
7    input String element;
8
9    output Real DXDT;
10  protected
11   constant Real rhoo = 881 "872 kg/m3";
12   constant Real rhow = 1064;
13   constant Real pi = Modelica.Constants.pi;
14   Real qin;
15   Real Ro;
16   Real Ri;
17   Real Vo_in;
18   Real ta;
19   Real rin;
20   Real FS;
21   Real r;
22   Real Rc;
23   Real vaaux "va used before apply the vancampen scaling factor";
24   Real va;
25   Real k;
26   Real vt;
27   Real rd;
28   Real Vo_c;
29   Real mum;
30   Real vr;
31
32  algorithm
33   qin := ini[1];
```

```
  Ro := ini[2];
  Ri := ini[3];
  Vo_in := ini[4];
  ta := ini[5];
  rin := ini[6];
  FS := ini[7];
  r := x;
  Rc := 0.25 * Ro;
  vaaux := qin / (pi * Ro ^ 2);
  //Van Campen thesis scaling factor Weak, Strong or Large
  if element == "w" then
    k := 3.5;
  elseif element == "s" then
    k := 5;
  elseif element == "l" then
    k := 7;
  end if;

  vt := k * vaaux;
  //Experimental correlation between rd and vt
  if vt > 4.45 then
    rd := ((-8 * vt) + 160) / 2 * 1e-6;
  else
    rd := ((-107 * vt) + 600) / 2 * 1e-6;
  end if;

   va := qin * (1 - FS) / (pi * (Ro ^ 2 - Ri ^ 2));
              Vo_c := Vo_in * (FS * (Ro ^ 2 - Ri ^ 2) * (Ro ^ 2 - r ^ 2) + (rin ^ 2 - Ri ^ 2) * (Ro ^ 2 -
 r ^ 2) * (1 - FS)) / ((r ^ 2 - Ri ^ 2) * (Ro ^ 2 - r ^ 2) * (1 - FS) + FS * (Ro ^ 2 -
 Ri ^ 2) * (Ro ^ 2 - r ^ 2));
   mum := 0.47 * Vo_c ^ 3 - 0.4 * Vo_c ^ 2 + 0.11 * Vo_c + 0.001;
   vr := (2 / 9 * (rhoo - rhow) * rd ^ 2 / mum * (vt * exp(-
0.04 * va * t / (2 * Ro))) ^ 2 / r * heaviside(r - Rc) + 2 / 9 * (rhoo -
 rhow) * rd ^ 2 / mum * r * (vt * exp(-0.04 * va * t / (2 * Ro)) / Rc) ^ 2 * heaviside(Rc -
 r)) * heaviside(r - 0.9 * Ri);
   DXDT := vr;

end swirl_sep2;
```

A.3.2 heaviside

This function was created to complete the code due to it is not implemented in the Modelica standard library.

```
function heaviside

  input Real heaviin;
  output Real heaviout;

algorithm
  if heaviin < 0 then
    heaviout := 0;
  elseif heaviin > 0 then
    heaviout := 1;
  else
    heaviout := 0.5;
  end if;

end heaviside;
```

A.3.3 DeOiler

The function estimates the outlet composition.

```
function DeOiler

  input Real qin;
  input Real Vo_in;
  input Real FS;
  input Real rin;
  input Real p2[3];
  output Real Vo_LPO;
  output Real Vo_HPO;
  output Real qen;
protected
  constant Real pi = Modelica.Constants.pi;
  constant Real k = 2e-4 ;
  Real Ro;
  Real Ri;
  Real qi;
  Real qo;
  Real u_HPO;
```

```
  Real u_LPO;
  Real du;

algorithm
  Ro := p2[2];
  Ri := p2[3];
  qi := FS * qin;
  qo := qin - qi;
  Vo_LPO := Vo_in * (FS * (Ro ^ 2 - Ri ^ 2) + (rin ^ 2 - Ri ^ 2) * (1 - FS)) / (FS * (Ro ^ 2 -
 Ri ^ 2));
  Vo_HPO := (Vo_in * qin - Vo_LPO * qi) / qo;
  u_HPO := qo / (pi * (Ro ^ 2 - Ri ^ 2));
  u_LPO := qi / (pi * Ri ^ 2);
  du := u_LPO - u_HPO;
  qen := k * du;
  if du >= 0 then
    Vo_LPO := (Vo_LPO * (qi - qen) + Vo_HPO * qen) / qi;
  else
    Vo_HPO := (Vo_HPO * (qo + qen) - Vo_LPO * qen) / qo;
    Vo_LPO := (Vo_in * qin - Vo_HPO * qo) / qi;
  end if;

  if Vo_LPO > 1 then
    Vo_LPO := 1;
  elseif Vo_LPO < Vo_in then
    Vo_LPO := Vo_in;
  end if;

  Vo_HPO := (Vo_in * qin - Vo_LPO * qi) / qo;

end DeOiler;
```

A.3.4 swirl_func3

This is the main function of Deoiler, the function estimates the outlet streams given the inlet conditions.

```
function swirl_func3

  import Oil_Water_Separation_System;
  input Real Vo_in;
  input Real qin;
  input Real FS;
  input String element;
```

```
8    output Real Vo_LPO;
9    output Real Vo_HPO;
10   output Real Edil;
11   output Real Edis;
12   output Real qen;
13 protected
14   constant Real pi = Modelica.Constants.pi;
15   //Parameters
16   parameter Real Lsw = 1.7;
17   parameter Real Ro = 0.05;
18   parameter Real Ri = 0.025;
19   //Local variables
20   Real qi;
21   Real qo;
22   Real ta;
23   Real rin;
24   Real x0;
25   Real ini[7];
26   Real h;
27   Real T[11];
28   Real X[11];
29   Real TotSep;
30   Real rin_0;
31   Real rho;
32   Real tol;
33   Real p2[3];
34
35 algorithm
36   qi := FS * qin;
37   qo := qin - qi;
38   ta := pi * (Ro ^ 2 - Ri ^ 2) * Lsw / qo;
39   rin := 0.999 * Ro;
40   x0 := rin;
41   ini := {qin, Ro, Ri, Vo_in, ta, rin, FS};
42   h := ta / 10;
43   (T,X) := RK2 ( {0.0,ta}, x0, h, ini,element);
44
45   if X[end] > Ri and X[end] < Ro then
46     TotSep := 0;
47     rin_0 := 0.99 * Ro;
48     rho := 1;
49     tol := 10 ^ (-10) * Ri;
50     (T,X) := shooting( {0.0,ta},h,Ri,rin_0,rho,tol,ini, element);
51     rin := X[1];
52   else
```

```
53    TotSep := 1;
54    rin := Ro;
55   end if;
56
57   p2 := {Lsw, Ro, Ri};
58   (Vo_LPO,Vo_HPO,qen) := DeOiler(qin, Vo_in, FS, rin, p2);
59   Edil := Vo_LPO * qi / (Vo_in * qin);
60   Edis := 1 - ((1 - Vo_LPO) * qi + Vo_HPO * qo) / qin;
61
62  end swirl_func3;
```

A.3.5 Block_Deoiler

This is the model that enables the simulation of Deoiler giving the inlet conditions to the rest of the functions.

```
1   model Block_Deoiler
2
3    import SI = Oil_Water_Separation_System.Units;
4
5    parameter SI.FlowSplit FS = 0.15 "Deoiler Flow split qLPO respect qinlet";
6    parameter String element = "l";
7
8     SI.Efficiency Edil
9      "Dilute efficiency, Fraction of oil that is kept in LPO, '1' if we don´t have oil in HPO, '0' if
10   we don´t have oil in LPO";
11    SI.Efficiency Edis
12     "Dispersed efficiency,how much of the liquid that exits through the desired outlet; '1'
13  means HPO is pure water and LPO is pure oil";
14    SI.OilVolumeFraction  qen
15     "Amount of reentrainment, Some water  will be re-
16  entrained in the light phase product and viceversa for low Flow Split, caused for the diffe
17  rence of axial velocities between LPO and HPO";
18
19   Oil_Water_Separation_System.Connector.Flow_port flowportindeo;
20   Oil_Water_Separation_System.Connector.Flow_port flowportoutdeoH;
21   Oil_Water_Separation_System.Connector.Flow_port flowportoutdeoL;
22
23  equation
24    (flowportoutdeoL.volfracc,flowportoutdeoH.volfracc, Edil, Edis, qen) = swirl_func3( flo
25  wportindeo.volfracc, flowportindeo.volflow, FS, element);
26     flowportoutdeoL.volflow =  flowportindeo.volflow*FS;
27    flowportoutdeoH.volflow =  flowportindeo.volflow*(1-FS);
```

```
28
29  end Block_Deoiler;
```

A.4 Dewaterer (Oil continuous phase)

A.4.1 swirl_sep2_o

This function contains the most part of the Deoiler model equations, returns the radial velocity of a droplet.

```
1   function swirl_sep2_o
2
3    input Real t;
4    input Real x;
5    input Real ini[:];
6    //Array size defined at function call time
7   input String element;
8    output Real vr;
9   protected
10   constant Real rhoo = 881 "872 kg/m3";
11   constant Real rhow = 1064;
12   constant Real beta = 10;
13   constant Real pi = Modelica.Constants.pi;
14   Real r;
15   Real qin;
16   Real Ro;
17   Real Ri;
18   Real Vo_in;
19   Real ta;
20   Real rin;
21   Real FS;
22   Real Rc;
23   Real va;
24   Real Vw_in;
25   Real vt0;
26   Real Vw;
27   Real mum;
28   Real f2;
29   Real f1;
30   Real f;
31   Real ac;
32   Real k;
33   Real rd;
```

```
algorithm
  qin := ini[1];
  Ro := ini[2];
  Ri := ini[3];
  Vo_in := ini[4];
  ta := ini[5];
  rin := ini[6];
  FS := ini[7];
  Vw_in := 1 - Vo_in;
  r := x;
  Rc := 0.25 * Ro;
  va := qin / (pi * Ro ^ 2);

  if element == "w" then
    k := 3.5;
  elseif element == "s" then
    k := 5;
  elseif element == "l" then
    k := 7;
  end if;

  vt0 := k * va;
  //Experimental correlation between rd and vt

  if vt0 > 4.45 then
    rd := ((-8 * vt0) + 160) / 2 * 1e-6;
  else
    rd := ((-107 * vt0) + 600) / 2 * 1e-6;
  end if;

  Vw := Vw_in * (((1 - FS) * Ri ^ 2 + FS * (Ri ^ 2 - rin ^ 2)) / ((1 - FS) * Ri ^ 2 + FS * (Ri ^ 2 -
 r ^ 2)));
  mum := 0.203 * Vw ^ 3 + 0.237 * Vw ^ 2 - 0.014 * Vw + 0.0088;
  f2 := (vt0 * exp(-0.04 * va * t / (2 * Ro))) ^ 2 / r;
  f1 := (vt0 * exp(-0.04 * va * t / (2 * Ro)) / Rc) ^ 2 * r;
  f := f2 - f1;
  ac := f2 - 0.5 * ((f ^ 2 + beta ^ 2) ^ 0.5 + f);
  vr := 2 / 9 * (rhow - rhoo) * rd ^ 2 / mum * ac;

end swirl_sep2_o;
```

A.4.2 DeWaterer

The function estimates the outlet composition.

```
function DeWaterer

  input Real qin;
  input Real Vo_in;
  input Real FS;
  input Real rin;
  input Real p3[3];
  output Real Vo_LPO;
  output Real Vo_HPO;
  output Real qen;
protected
  constant Real pi = Modelica.Constants.pi;
  constant Real k = 2e-4 ;
  Real Ro;
  Real Ri;
  Real qi;
  Real qo;
  Real Vw_LPO;
  Real Vw_HPO;
  Real u_HPO;
  Real u_LPO;
  Real du;

algorithm
  Ro := p3[2];
  Ri := p3[3];
  qi := FS * qin;
  qo := qin - qi;
  Vw_HPO := (1 - Vo_in) * (((1 - FS) * Ri ^ 2 + (Ri ^ 2 - rin ^ 2) * FS) / ((1 - FS) * Ri ^ 2));
  Vw_LPO := ((1 - Vo_in) * qin - Vw_HPO * qo) / qi;
  u_HPO := qo / (pi * (Ro ^ 2 - Ri ^ 2));
  u_LPO := qi / (pi * Ri ^ 2);
  du := u_LPO - u_HPO;
  qen := k * du;

  if du >= 0 then
    Vw_LPO := (Vw_LPO * (qi - qen) + Vw_HPO * qen) / qi;
    Vw_HPO := ((1 - Vo_in) * qin - Vw_LPO * qi) / qo;
  else
    Vw_HPO := (Vw_HPO * (qi + qen) - Vw_LPO * qen) / qi;
```

```
41    end if;
42
43    if Vw_HPO > 1 then
44      Vw_HPO := 1;
45    elseif Vw_HPO < 1 - Vo_in then
46      Vw_HPO := 1 - Vo_in;
47    end if;
48
49    Vw_LPO := ((1 - Vo_in) * qin - Vw_HPO * qo) / qi;
50    Vo_LPO := 1 - Vw_LPO;
51    Vo_HPO := 1 - Vw_HPO;
52
53  end DeWaterer;
```

A.4.3 swirl_func3_o

This is the main function of Dewaterer, the function estimates the outlet streams given the inlet conditions.

```
1   function swirl_func3_o
2
3     import Oil_Water_Separation_System;
4     input Real Vo_in;
5     input Real qin;
6     input Real FS;
7     input String element;
8     output Real Vo_LPO;
9     output Real Vo_HPO;
10    output Real Edil;
11    output Real Edis;
12    output Real qen;
13  protected
14    constant Real pi = Modelica.Constants.pi;
15    //Parameters
16    parameter Real Lsw = 1.7;
17    parameter Real Ro = 0.05;
18    parameter Real Ri = 0.043;
19    //Local variables
20    Real qi;
21    Real qo;
22    Real ta;
23    Real rin;
24    Real ini[7];
25    Real h;
```

```
  Real T[11];
  Real X[11];
  Real Vw_LPO;
  Real Vw_HPO;
  Real rin_0;
  Real rho;
  Real tol;
  Real p3[3];

algorithm
  qi := FS * qin;
  qo := qin - qi;
  ta := pi * Ri ^ 2 * Lsw / qi;
  rin_0 := 0.8 * Ri;
  ini := {qin, Ro, Ri, Vo_in, ta, rin_0, FS};
  h := ta / 10;
  rho := 1;
  tol := 10 ^ (-10) * Ri;
  (T, X) := Shooting ({0.0, ta}, h, Ri, rin_0, rho, tol, ini, element);
  rin := X[1];
  p3 := {Lsw, Ro, Ri};
  (Vo_LPO,Vo_HPO,qen) := DeWaterer(qin,Vo_in,FS,rin,p3);
  Vw_LPO := 1 - Vo_LPO;
  Vw_HPO := 1 - Vo_HPO;
  Edil := Vo_LPO * qi / ( Vo_in * qin);
  Edis := 1 - ((1 - Vo_LPO) * qi + Vo_HPO * qo) / qin;
end swirl_func3_o;
```

A.4.4 Block_Dewaterer

This is the model that enables the simulation of Dewaterer giving the inlet conditions to the rest of the functions.

```
model Block_Dewaterer
  import Oil_Water_Separation_System;
   import SI = Oil_Water_Separation_System.Units;
  parameter SI.FlowSplit FS = 0.91 "Dewaterer Flow split qLPO respect qinlet";
  parameter String element = "l";

   SI.Efficiency Edil
    "Dilute efficiency, Fraction of oil that is kept in LPO, '1' if we don´t have oil in HPO, '0' if
 we don´t have oil in LPO";
   SI.Efficiency Edis
    "Dispersed efficiency,how much of the liquid that exits through the desired outlet; '1'
```

```
means HPO is pure water and LPO is pure oil";
  SI.OilVolumeFraction  qen
   "Amount of reentrainment, Some water  will be re-
entrained in the light phase product and viceversa for low Flow Split, caused for the diffe
rence of axial velocities between LPO and HPO";

 Oil_Water_Separation_System.Connector.Flow_port flowportindew;
 Oil_Water_Separation_System.Connector.Flow_port flowportoutdewH;
 Oil_Water_Separation_System.Connector.Flow_port flowportoutdewL;
algorithm
 (flowportoutdewL.volfracc,flowportoutdewH.volfracc,Edil,Edis,qen) :=swirl_func3_o(
   flowportindew.volfracc, flowportindew.volflow, FS,element);

  flowportoutdewL.volflow := flowportindew.volflow*FS;
  flowportoutdewH.volflow := flowportindew.volflow*(1-FS);

end Block_Dewaterer;
```

A.5 Gravity separator

The function returns the bottom and top outlet flow rates and oil volume fractions given the inlet conditions and flow split of the separator.

```
model Block_GravitySeparator

 import SI = Oil_Water_Separation_System.Units;
 //Constants
 constant Real pi = Modelica.Constants.pi;
 constant Real g = Modelica.Constants.g_n;
 //Parameters
 parameter SI.Length Lsep = 7.0 "Gravity separator length";
 parameter SI.Length R = 1.7 "Separator radio";
 parameter SI.Density rhoo = 881 "Oil Density";
 parameter SI.Density rhow = 1064 "Water Density";
 parameter SI.Length rd = 60e-6 "Droplet Radio";
 parameter SI.FlowSplit FS = 0.33 "Flow split of the gravity separator";
 parameter SI.VolumeFlowRate qin = 0.006944 "Inlet flow rate";
 parameter SI.OilVolumeFraction vo_in = 0.4 "Inlet oil volume fraction";
 //Variables
 SI.VolumeFlowRate qt "Flow to dewatered";
 SI.VolumeFlowRate qb "Flow to deoiler";
```

```
  SI.Length Hw "Wall heigth";
  SI.DynamicViscosity mum "Emmulsion dynamic viscosity ";
  SI.Area AHw "cross section area of the lower part of the separator(the circular
segment limited by wall height)";
  SI.Velocity vh
    "Horizontal velocity of a droplet moving through the separator under wall height";
  SI.Velocity vv "Vertical velocities of the droplets caused by the gravitational buoyancy
      forces";
  SI.Length h "vertical distance, of a droplet entering the separator at the bottom
of the tank, travels during its residence time in the separator";
  SI.Length d "Distance between wall and h";
  SI.Area Ad "Area of this circular segment";
  SI.OilVolumeFraction vw_b "Water volume fraction of the bottom outlet";
  SI.OilVolumeFraction vo_b "Oil volume fraction of the bottom outlet";
  SI.OilVolumeFraction vo_t "Oil volume fraction of the top outlet";
  SI.OilVolumeFraction vo_bn "Auxiliar value";
  //Ports
 Oil_Water_Separation_System.Connector.Flow_port flowporttop "Outlet top part";
 Oil_Water_Separation_System.Connector.Flow_port flowportbottom
    "Outlet bottom part";
  //equations

equation
  Hw = 0.75 * 2 * R;
  qt = FS * qin;
  qb = qin - qt;
  mum = 0.47 * vo_in ^ 3 - 0.4 * vo_in ^ 2 + 0.11 * vo_in + 0.001;
  AHw = R ^ 2 / 2 * (2 * acos((R - Hw) / R) - sin(2 * acos((R - Hw) / R)));
  vh = qb / AHw;
  vv = 2 * (-g) * (rhoo - rhow) * rd ^ 2 / (9 * mum);
  h = Lsep * (vv / vh);
  d = max(Hw - h, 0) "%if h>Hw -> vo_b=0";
  Ad = R ^ 2 / 2 * (2 * acos((R - d) / R) - sin(2 * acos((R - d) / R)));
  vw_b = (AHw - Ad) / AHw + Ad / AHw * (1 - vo_in);
  vo_bn = 1 - vw_b;
  vo_t = min((vo_in - vo_bn * (1 - FS)) / FS, 1);

  if vo_t >= 1 then
    vo_b = (vo_in - vo_t * FS) / (1 - FS);
  else
    vo_b = vo_bn;
  end if;

  //Set the ports
  flowporttop.volflow = qt;
```

```
64    flowportbottom.volflow = qb;
65     flowporttop.volfracc = vo_t;
66    flowportbottom.volfracc = vo_b;
67
68  end Block_GravitySeparator;
```

A.6 Node

The node connect two inputs mixing them and given the outlet flow rate and oil volumetric fraction.

```
1   model Node_2inlets_1outlet
2
3   Oil_Water_Separation_System.Connector.Flow_port flowportinleta;
4   Oil_Water_Separation_System.Connector.Flow_port flowportinletb;
5   Oil_Water_Separation_System.Connector.Flow_port flowportoutlet;
6
7   equation
8    flowportoutlet.volflow =  flowportinleta.volflow + flowportinletb.volflow;
9    flowportoutlet.volfracc =  ((flowportinleta.volflow*flowportinleta.volfracc) + (flowporti
10  nletb.volflow*flowportinletb.volfracc))/(flowportinleta.volflow + flowportinletb.volflow);
11
12  end Node_2inlets_1outlet;
```

A.7 Numerical solvers

A.7.1 Runge-Kutta

The function is a second order Runge-Kutta solver, is called by swirl_func3 and swirl_func3_o to solve the radial inlet position of the droplet.

```
1   function RK2
2
3    input Real tspan[:];
4    input Real yi;
5    input Real h;
6    input Real ini[:];
7    input String element;
8    output Real t[11];
9    output Real y[11];
```

```
protected
  Real d;
  Real k1;
  Real k2;
  Integer i;
  //Real aux;

algorithm
  t := tspan[1]:h:tspan[2];

  while t[end] < tspan[2] loop
    t[end + 1] := tspan[2];
  end while;

  //aux := size(t, 1) - 1;
  d := t[2] - t[1];
  y[1] := yi;
  for i in 1:10 loop
    k1 := d * swirl_sep2_o(t[i], y[i], ini, element);
    k2 := d * swirl_sep2_o(t[i + 1], y[i] + k1, ini, element);
   // k1 := d*swirl_sep2(t[i],y[i],ini,element) change in dewaterer;
   // k2 := d*swirl_sep2(t[i + 1],y[i] + k1,ini,element) change in dewaterer;

    y[i + 1] := y[i] + (k1 + k2) / 2;
  end for;

end RK2;
```

A.7.2 Shooting method

The shooting method (Newton-Raphson method) solve the value boundary problem finding the droplet inlet position.

```
function shooting

  import Oil_Water_Separation_System;
  input Real tspan[:];
  input Real h;
  input Real yf;
  input Real gamma0;
  input Real rho;
  input Real tol;
  // Ini and element are the parameters needed to call swirl_sep2
  input Real ini[:];
```

```
12    input String element;
13    output Real t[11];
14    output Real y[11];
15  protected
16    Real maxiter;
17    Real gammanew;
18    Real iter;
19    Real fnk;
20    Real gamma1;
21    Real dgamma;
22    Real a;
23    Real fnka;
24    Real jacob;
25    Real init[7];
26    Real ta[11];
27    Real ya[11];
28
29  algorithm
30    init[:] := ini;
31    gammanew := gamma0;
32    iter := 0;
33    maxiter := 100;
34    fnk := 10 * yf;
35
36    while abs(yf - fnk) > tol and iter < maxiter loop
37      iter := iter + 1;
38      gamma1 := gammanew;
39      init[6] := gamma1;
40      (t,y) := RK2( tspan, gamma1,h,init,element);
41      fnk := y[end];
42
43      if gamma1 <> 0 then
44        dgamma := -gamma1 / 100;
45      else
46        dgamma := -0.01;
47      end if;
48
49      a := gamma1 + dgamma;
50      init[6] := a;
51      (ta,ya) := RK2( tspan, a,h, init,element);
52      fnka := ya[end];
53      jacob := (fnka - fnk) / dgamma;
54      a := gamma1 - dgamma;
55
56      if jacob == 0 then
```

```
57        gammanew := gamma1 + max(abs(dgamma), abs(1.1 * tol));
58      else
59        gammanew := gamma1 - rho * (1 / jacob) * (fnk - yf);
60      end if;
61
62    end while;
63
64  end shooting;
```