2d Optimizaton WCF Service Overview

Version 1.0 (alfa)

Optimizing the cutting of sheet materials.
The service allows to perform the layout of blanks on sheets of rectangular shape. It takes into account the technological parameters in order to obtain a minimum percentage of material waste.
Used guillotine cutting (https://en.wikipedia.org/wiki/Guillotine_problem)

Scenarios

1. Loading and storing documents for optimization. Optimization of cutting with different parameters. Changing the optimization settings.
2. Getting optimization results in a standard report (PDF)
3. The ability to create custom reports
4. Unloading details about cutting in the form of cuts and their coordinates (XML)
5. Unloading files on the cutting tables (Hegla, Bystronic, Lisec, Intermac)

Background

Security

When registering, the service address and secret key are provided, which is used later for authorization.

Configuring for .net project

1) Add Service reference to you project
2) Add code for create client

```
ServiceReference1.SquareOptServiceClient getClient() {
    var timeSpan = new TimeSpan(0, 30, 0);
    string uri = "http://<IP ADDRESS>:<PORT>/SquareOptService.svc";

    var binding = new BasicHttpBinding();
    binding.Name = "BasicHttpBinding_IService";
    binding.ReceiveTimeout = timeSpan;
    binding.SendTimeout = timeSpan;
    binding.ReaderQuotas.MaxArrayLength = Int32.MaxValue;
    binding.MaxReceivedMessageSize = Int32.MaxValue;

    var address = new EndpointAddress(uri);

    var client = new ServiceReference1.SquareOptServiceClient(binding, address);
    return client;
}
```

Code Examples

Download sample project link (zip)
`companyCode` – Security key. For testing service use key: "test service"

1) Add new optimization

```
var client = getClient();
var doc = client.Add(companyCode);
```

2) Get list documents

```
var client = getClient();
var docs = client.GetHeaders(companyCode, new DateTime(2017, 01, 01), DateTime.Now);
```

3) Modify document

```
var client = getClient();
var doc = client.Load(companyCode, 12345 /*id document*/);
doc.Comment = «new comment»;
client.Save(companyCode, doc);
```

4) Optimizaton

```
var client = getClient();
var result = client.Optimize(companyCode, 12345 /*id document*/);
```

5) Get default PDF report

```
var client = getClient();
//function return byte[]
var result = client.Export(companyCode, 12345,
ServiceReference1.ExportMode.DefaultReportPdf, String.Empty);
```

Download sample PDF report this link.

6) Get data for Hegla

```
var client = getClient();
var heglaArgs = @"{"Description":"4 mm","Article":"4","Thickness":4}";
//function return byte[]
var result = client.Export(companyCode, 12345, ServiceReference1.ExportMode.Hegla,
heglaArgs);
```

Class Reference

```
public class Document {

    public int Id { get; set; }
    public DateTime Date { get; set; }
    public string Description { get; set; }
    public List<Plate> Plates { get; set; }
    public List<Destination> Destinations { get; set; }
    public int MaxLenCuttingX { get; set; }
    public int MaxLenCuttingY { get; set; }
    public CutDepth CutDepth { get; set; }
    public string Status { get; set; }
    public bool IsOptimized { get; set; }
    public int? DestinationCount { get; set; }
    public decimal? DestinationSquare { get; set; }
    public int? PlateCount { get; set; }
    public decimal? PlateSquare { get; set; }
    public int? LastDestinationWidth { get; set; }
    public int? LastDestinationHeight { get; set; }
    public decimal? WasteSquare { get; set; }
    public decimal? WasteWorkSquare { get; set; }
    public decimal? WasteTrashSquare { get; set; }
    public decimal? WastePercentA { get; set; }
    public decimal? WastePercentB { get; set; }

    public Document();
}
```

```
public enum CutDepth {
    Max,
    Z,
    W
}

public enum ExportMode {
    DefaultReportPdf,
    //CustomReportPdf,
    //Emf,
    //Xml,
    Hegla,
    //Bystronic,
    //Lisec,
    Intermac
}


public class Destination {
    public int NumPos { get; set; }
    public int Width { get; set; }
    public int Height { get; set; }
    public int Qu { get; set; }
    public bool IsRequared { get; set; }
    public int L { get; set; }
    public int T { get; set; }
    public int R { get; set; }
    public int B { get; set; }
    public int dCut { get; set; }
    public int CarryOverWidth { get; set; }
    public int WorkWidth { get; set; }
    public int WorkHeight { get; set; }
}


public class Plate {
    public int NumPos { get; set; }
    public int Width { get; set; }
    public int Height { get; set; }
    public int Qu { get; set; }
    public string Description { get; set; }
    public int Priority { get; set; }
}



public interface ISquareOptService {
    string Test();
    Document Add(string companyCode);
    Document[] GetHeaders(string companyCode, DateTime from, DateTime to);
    Document Load(string companyCode, int id);
    void Save(string companyCode, Document document);
    void Delete(string companyCode, int id);
    string[] Check(string companyCode, int id);
    bool Optimize(string companyCode, int id);
    byte[] Export(string companyCode, int id, ExportMode mode, string paramenets);
}
```