Membuat Workflow menggunakan K2 Studio for VS2010

Pendahuluan

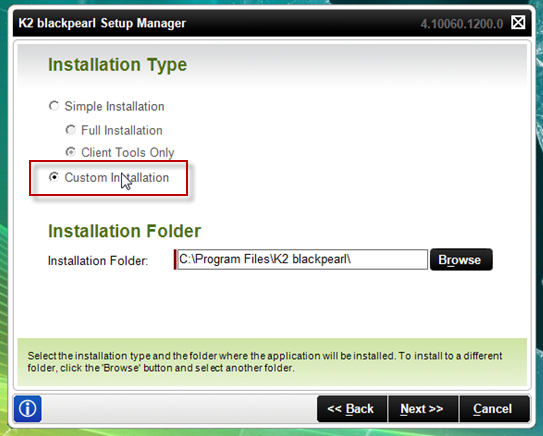
Di project BPM kita akan sering mendengar istilah business process. Business process ini pada dasarnya adalah workflow dari sebuah dokumen, dimulai dari pembuatan dokumen tersebut, lalu pengajuannya ke orang2 tertentu untuk mendapatkan approval, sampai akhirnya dokumen tersebut statusnya approved dan bisa dieksekusi.

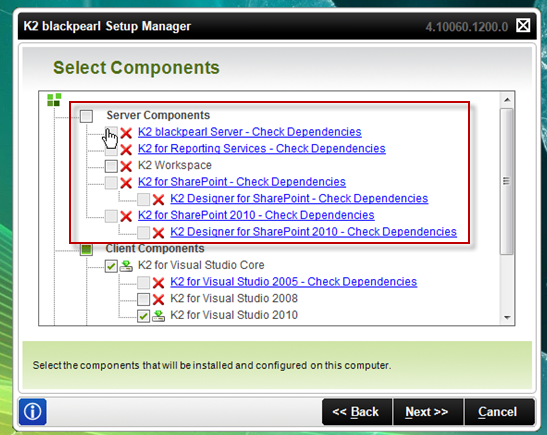
Business process ini dibuat menggunakan bantuan workflow engine, dalam hal ini kita menggunakan K2 Blackpearl sebagai workflow engine. K2 Blackpearl menggunakan Visual Studio dalam pembuatan sebuah workflow. Versi Visual Studio yang akan kita gunakan pada tutorial ini adalah Visual Studio 2010 (selanjutnya disingkat VS2010).

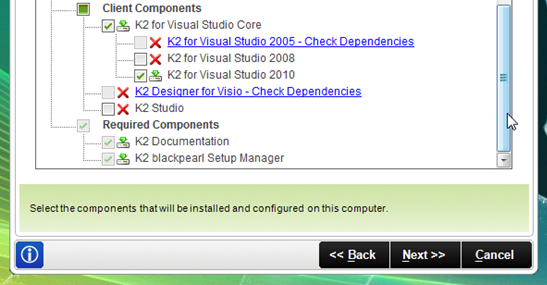
Instalasi

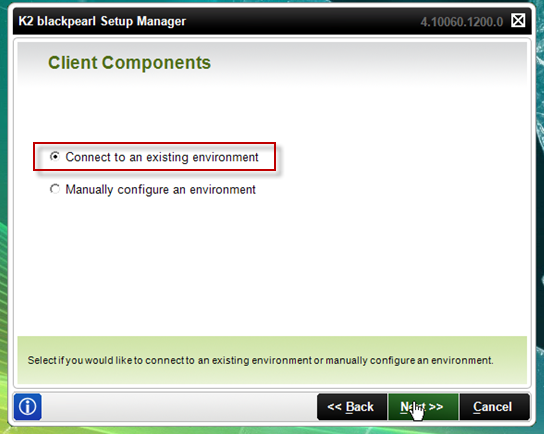
Untuk memulai membuat workflow menggunakan VS2010, kita harus menginstal salah satu komponen dari K2 Blackpearl, yaitu K2 for VS2010. Versi installer yang digunakan adalah 4.10060.1200.0. Kalau di komputer Anda sudah terinstal K2 for VS versi sebelumnya, maka harus di uninstall terlebih dahulu sebelum melakukan proses instalasi.

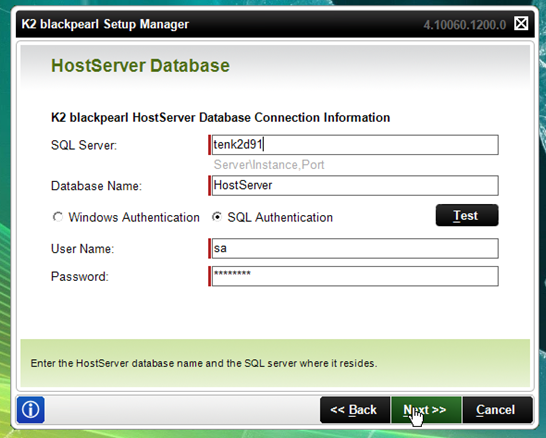
Berikut ini cara instalasinya :

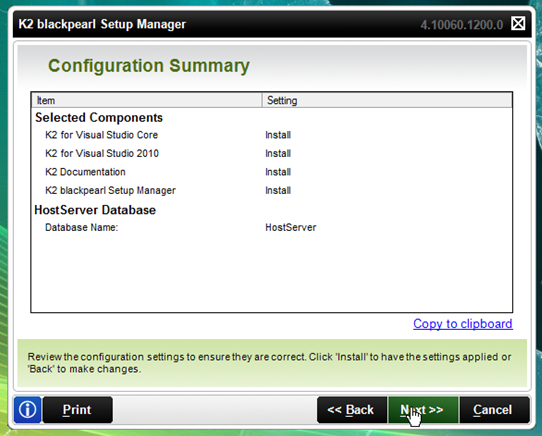


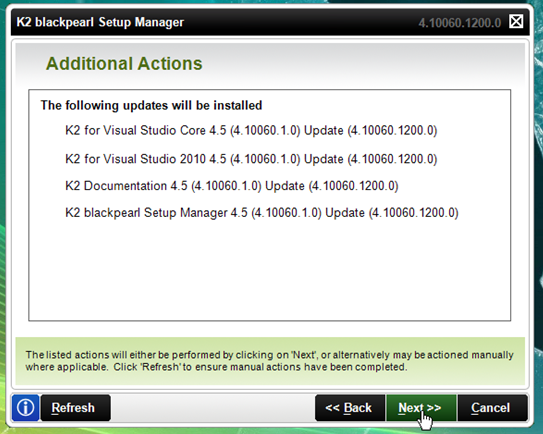






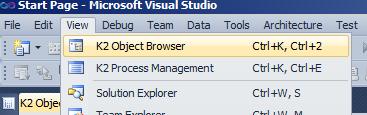




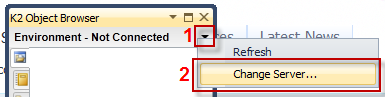


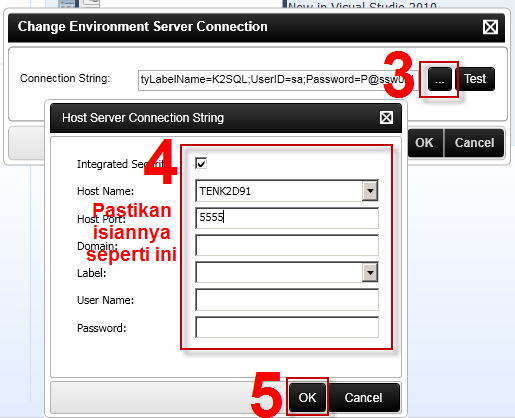
Troubleshooting setelah instalasi

Setelah instalasi, di Visual Studio akan ada beberapa menu baru, salah satunya ada di View > K2 Object Browser.

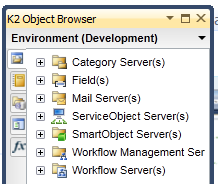


Bila di K2 Object Browser statusnya masih Not Connected, lakukan langkah2 berikut :





Hasilnya seharusnya seperti ini :



Membuat workflow sederhana : TDM - approval hingga manager L4

Contoh business process yang akan kita buat adalah business process untuk input data Technical Data Management (TDM).

Business processnya sebagai berikut : seorang operator di lapangan bertugas melakukan input data, kemudian data2 tersebut dikirim kepada atasannya secara bertahap hingga ke level manager L4. Setelah proses approval selesai, maka data tersebut dianggap sudah valid dan bisa ditampilkan pada report di dashboard TDM.

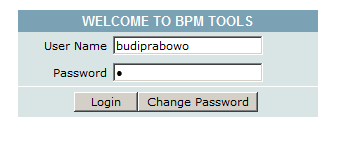
Sang manager selain bisa melakukan approval, beliau juga mempunyai pilihan untuk mengembalikan data tersebut ke operator atau ke approver sebelumnya jika dianggap ada data yang perlu diperbaiki.

Sekilas tentang struktur organisasi

Dalam penjelasan tentang business process TDM, terdapat istilah manager L4 (level 4). Untuk memahami apa yang dimaksud dengan manager L4, kita harus mengetahui struktur organisasi yang digunakan. Dalam struktur organisasi, terdapat user2 yang duduk dalam struktur manajerial. Dimulai dari yang tertinggi yaitu President Director atau disebut juga L1. Lalu di bawah L1 ada L2, di bawah L2 ada L3, demikian seterusnya hingga L6. Sedangkan user2 yang tidak duduk dalam manajerial tidak mempunyai level. Dalam contoh business process TDM, sang operator yang melakukan input data bukanlah seorang manager (tidak mempunyai level). Atasan langsung dari sang operator mungkin baru L5. Atasan dari L5 ini barulah seorang L4. Jadi dokumen TDM yang diinput oleh sang operator tersebut harus melalui approval dua kali (approval oleh L5 dan L4nya) agar statusnya approved. Itulah yang dimaksud dengan approval hingga L4.

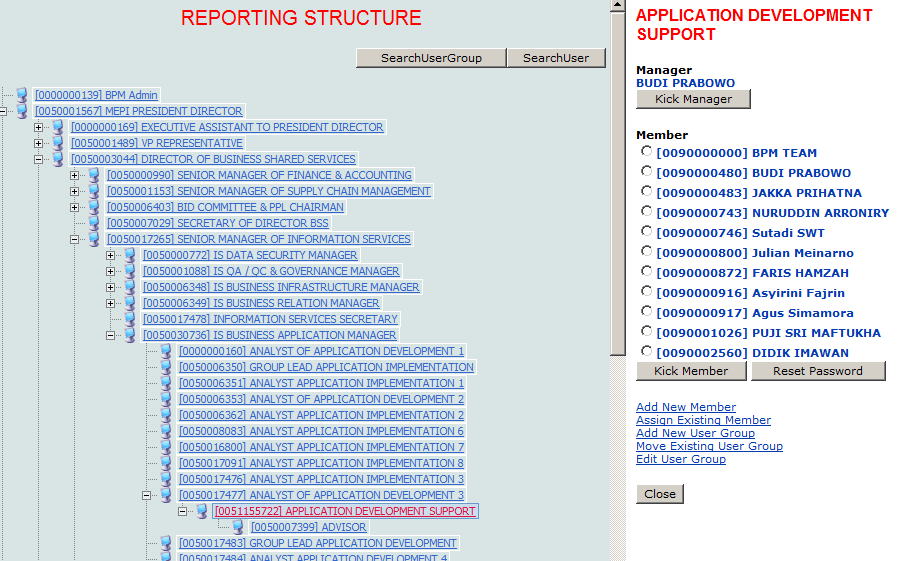
Dalam struktur organisasi, terdapat dua jenis entitas : user dan position. Hierarki struktur organisasi disusun berdasarkan posisi, sedangkan user hanya menjadi anggota dari posisi2 tersebut (tidak ada hierarki berdasarkan user secara individu). Satu posisi bisa memiliki banyak user, demikian juga satu user bisa mempunyai banyak posisi.

Untuk melihat hierarki struktur organisasi secara visual, bisa diakses melalui halaman berikut : <http://tenbpd91.medcoenergi.com/bpmtools/login.aspx>.



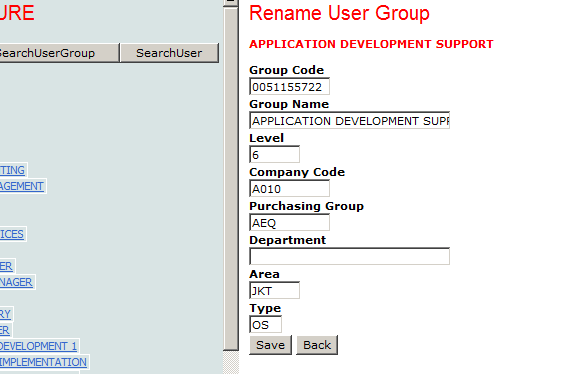


Reporting structure all :

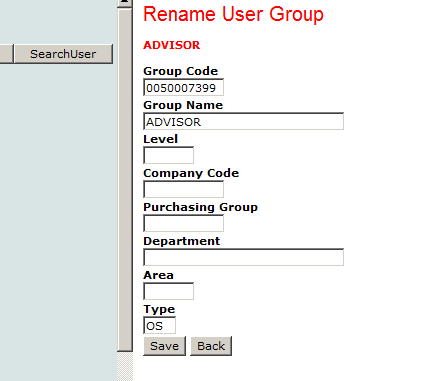


Edit User Group :

Position level 6



Position tanpa level :



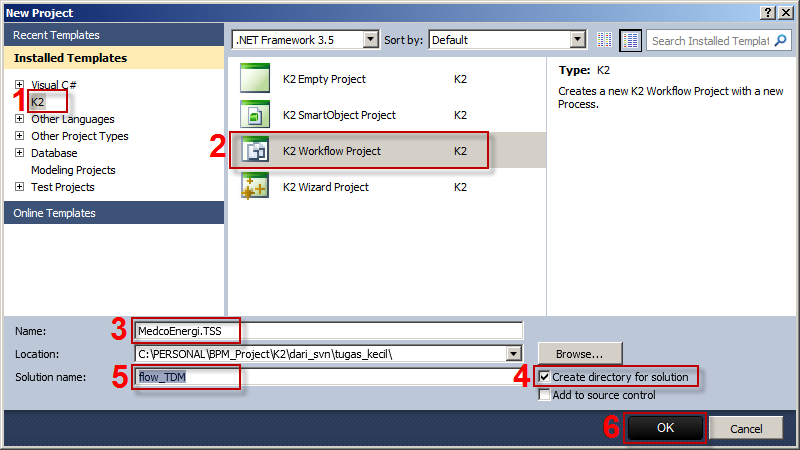
Back to topic : Create workflow for TDM scenario

Dalam teknis pembuatan sebuah workflow, sebenarnya kita akan memerlukan dua macam project di Visual Studio, yaitu :

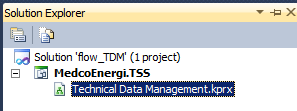
1. Sebuah WCF service library yang berfungsi sebagai ‘otak’ dari workflow. Semua logic yang berhubungan dengan business process akan di-coding di WCF service library ini.
2. Sebuah K2 Workflow project yang berfungsi sebagai ‘badan’ yang me-routing dokumen ke pihak-pihak yang sudah ditentukan. Coding utama di dalam K2 workflow ini hanyalah pemanggilan WCF service yang sudah dibuat. Sisanya adalah drag and drop komponen2 workflow yang sebagian besar dibantu oleh wizard.

Kita akan mulai dari yang mudah terlebih dahulu ☺, yaitu pembuatan K2 workflow project.

Silahkan buat sebuah K2 Workflow Project untuk mulai membuat workflow TDM dengan langkah2 sbb :



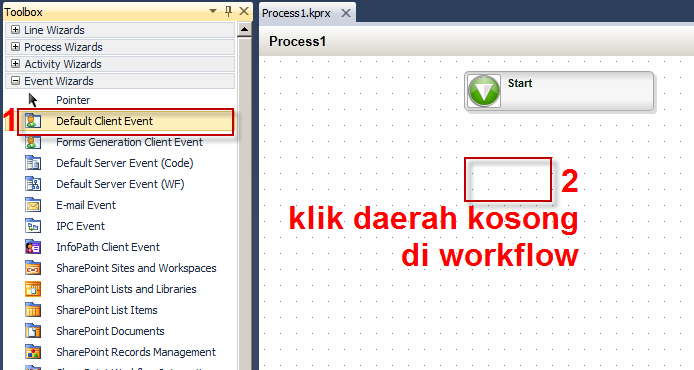
Dari solution explorer, rename nama workflownya menjadi Technical Data Management sbb :

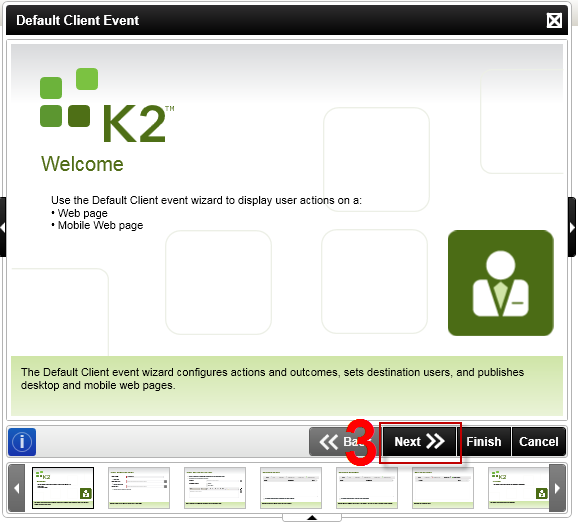


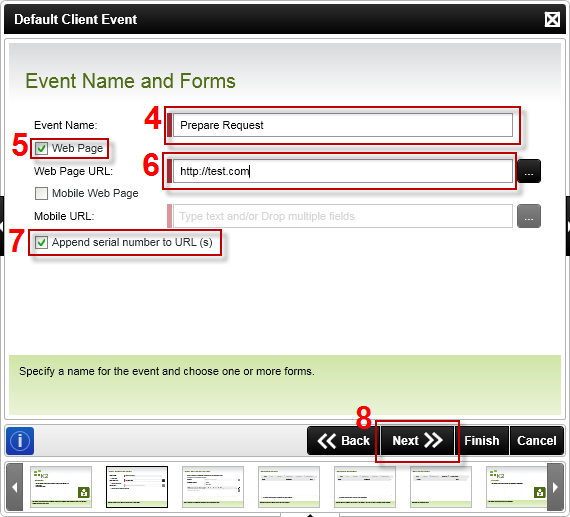
Kita mulai dengan membuat sebuah client event di mana sang operator bisa mengisikan data TDM. Client event adalah event pada workflow yang bisa dihubungkan dengan sebuah halaman web melalui sebuah URL. Melalui halaman web inilah si user participant dari client event memberikan respon.

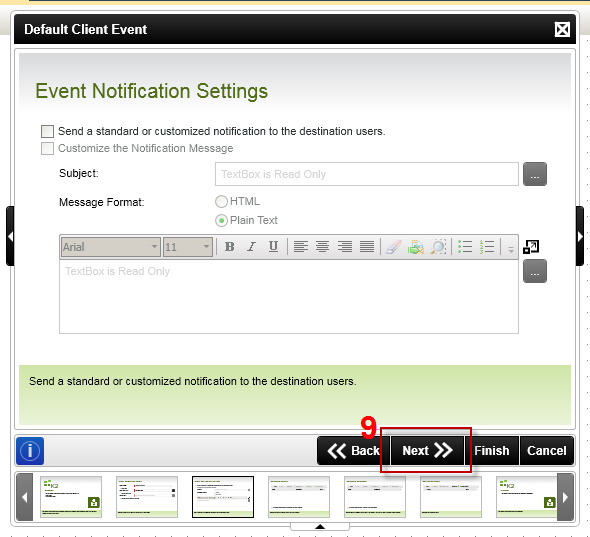
**Penting** : Dalam client event memang diperlukan sebuah URL ke halaman web dari form isian /approvalnya. Tetapi bukan berarti sebuah workflow tidak bisa dikerjakan sampai tuntas jika form isiannya belum selesai, karena yang dibutuhkan dari sisi workflow bukanlah halaman form itu sendiri, tetapi respon user lah yang sebenarnya diperlukan. Dan respon user ini bisa dilakukan tanpa melalui form. Lebih jauh tentang ini akan dibahas nanti dalam bagian testing workflow.

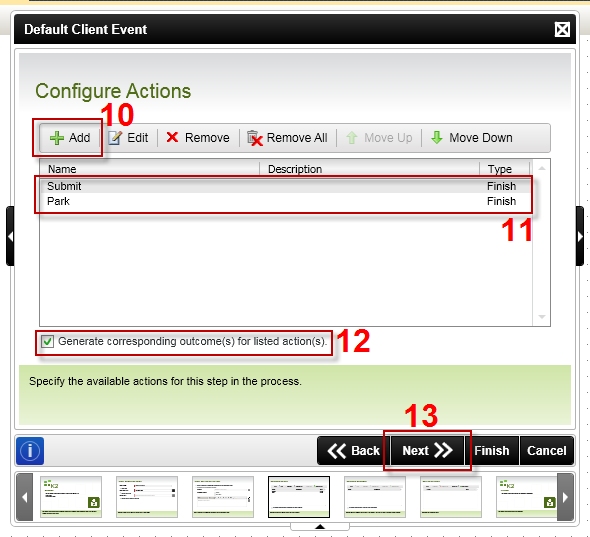
Berikut ini cara menambahkan client event :

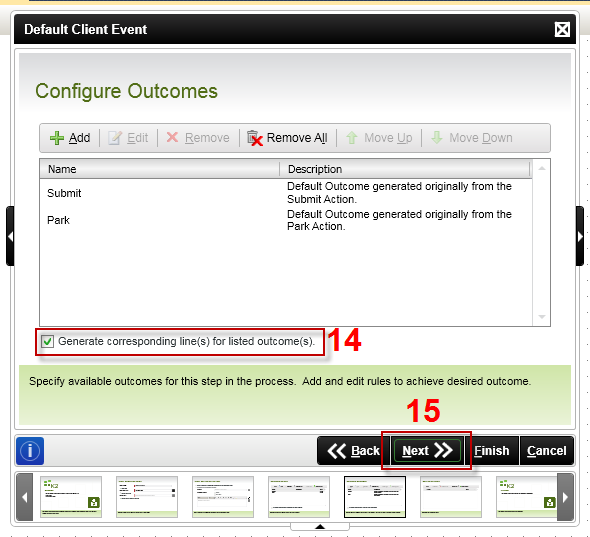


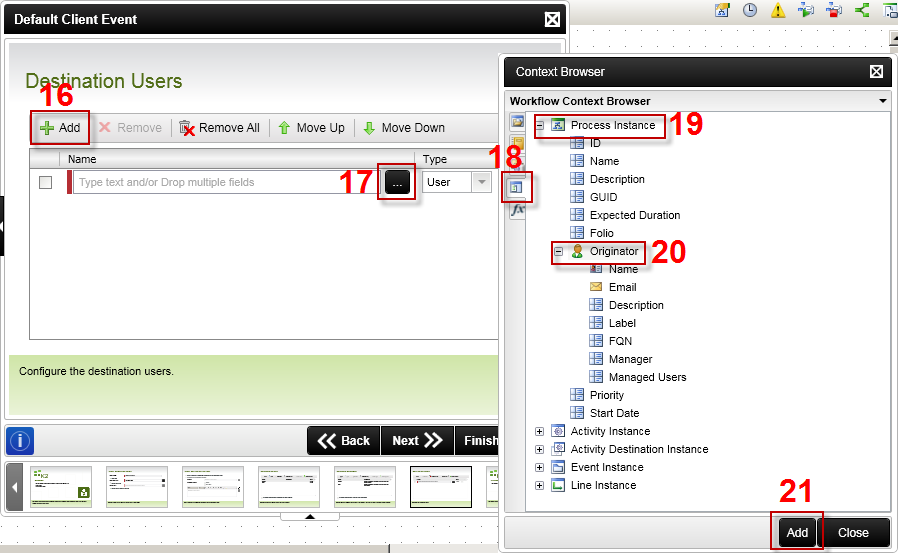


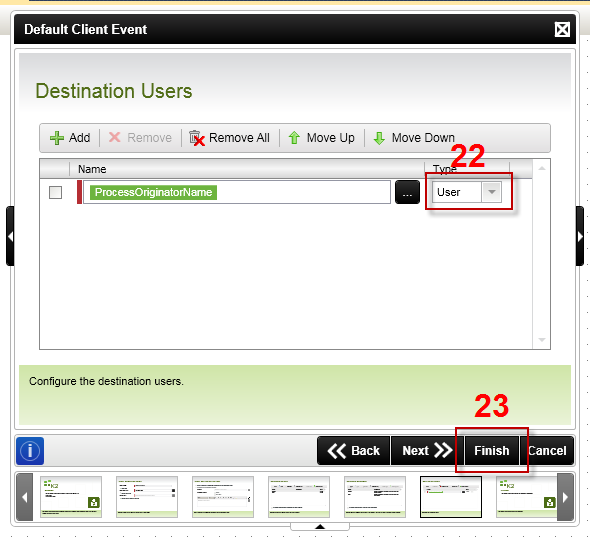


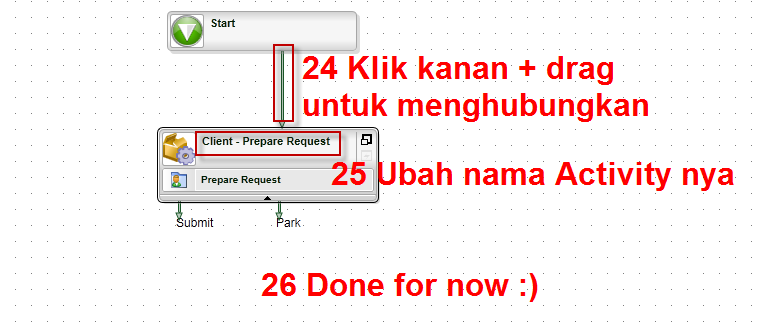




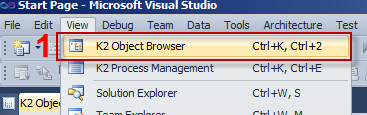


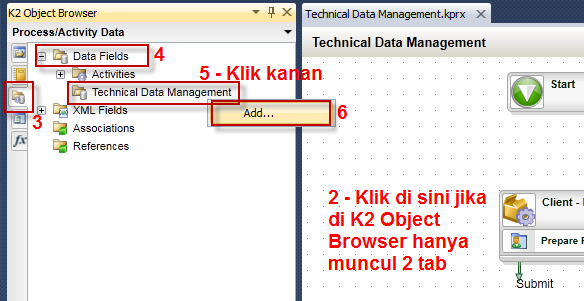


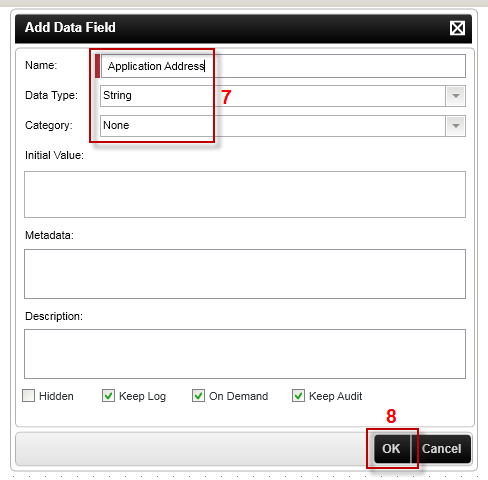




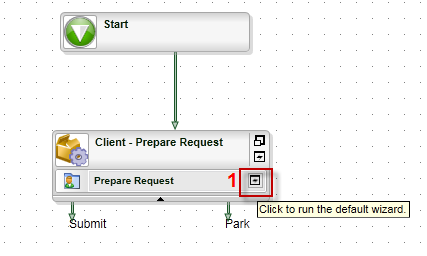
Di langkah nomor 6 penambahan client event prepare request, kita masih mengisikan sebuah URL dummy secara hard code (http://test.com) sebagai form isiannya. Nantinya, URL ini akan kita ubah sehingga tidak lagi di hard code, tetapi mengambil dari sebuah variabel, atau dalam istilah K2 disebut Data Fields. Sekarang kita akan menambahkan sebuah Data Fields bernama Application Address pada workflow TDM sbb :

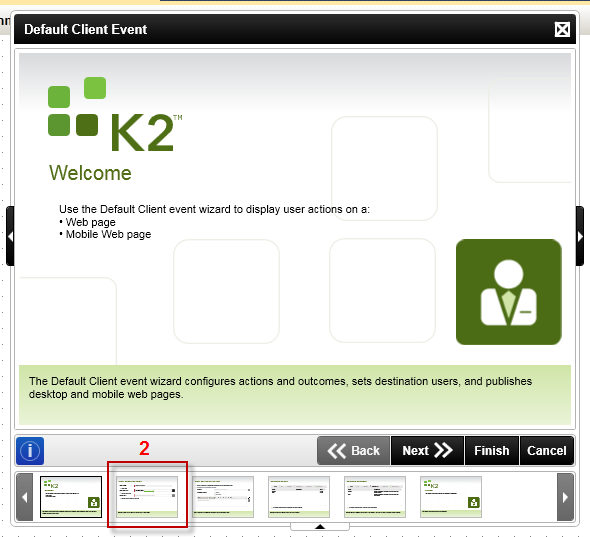


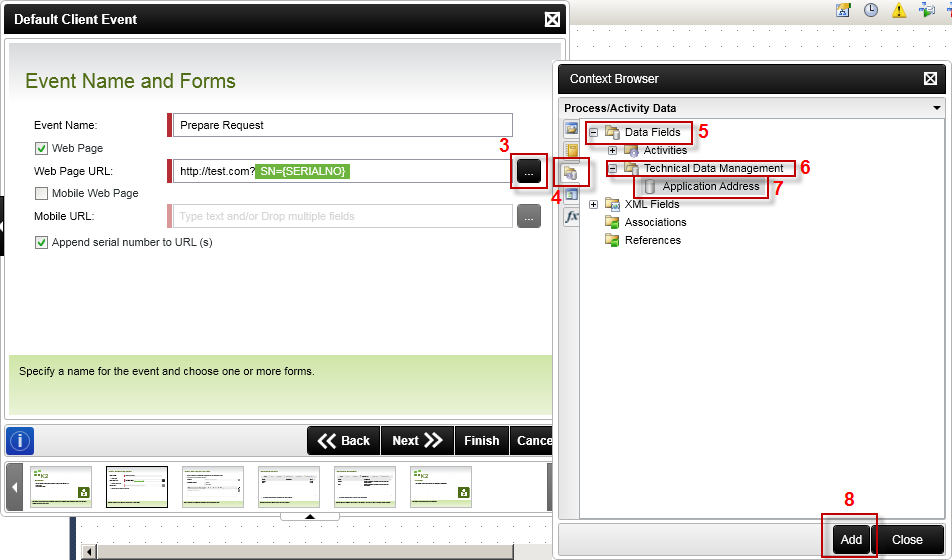


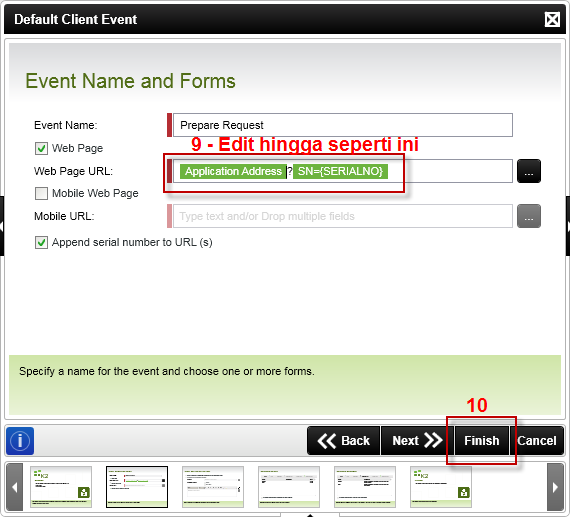


Sekarang kita ganti URL yang di hard code tadi dengan Data Field yang barusan kita tambahkan sbb :





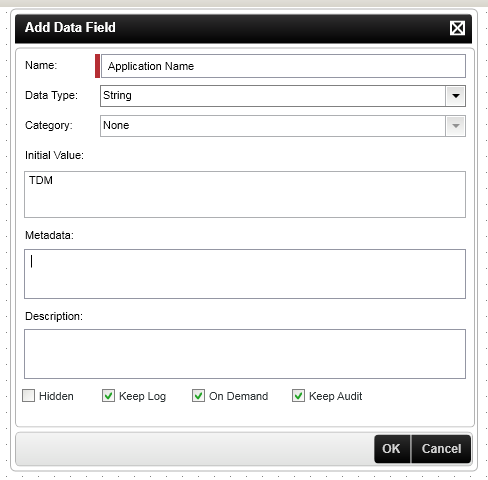


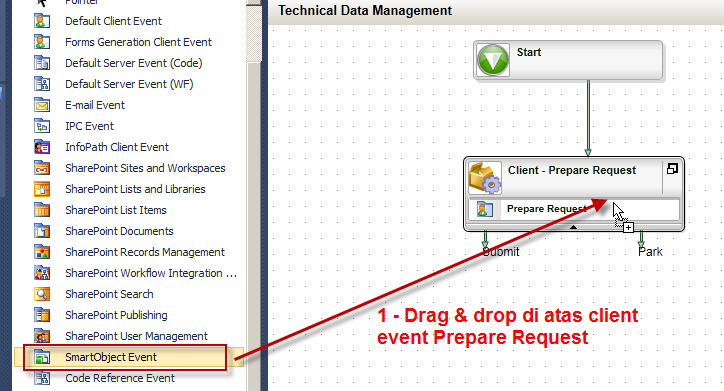


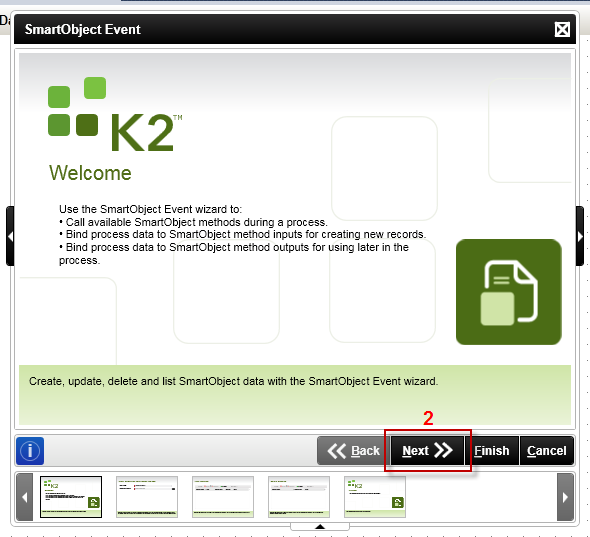
Sekarang URL dari form sudah tidak di hard-code lagi, tetapi sudah mengambil dari nilai variabel Application Address.

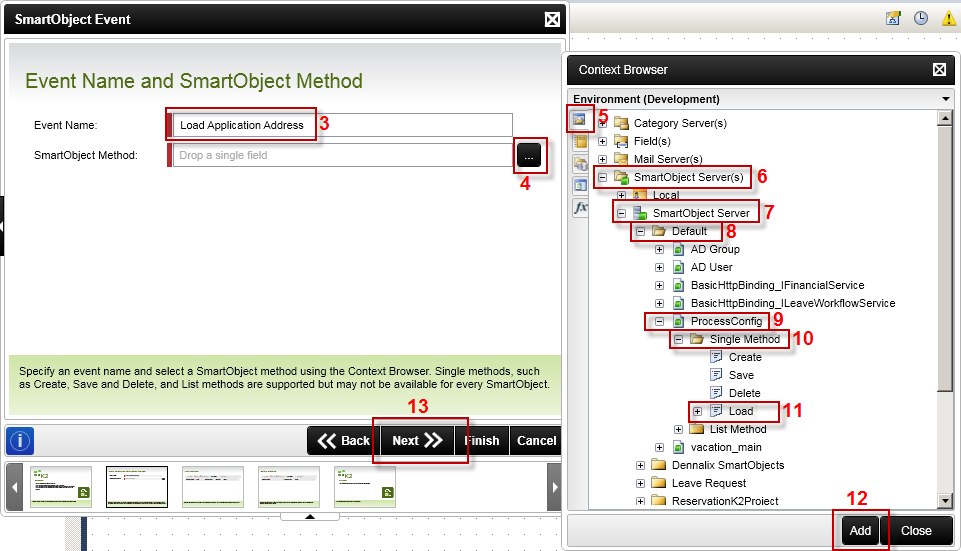
Ketika menambahkan data field Application Address ini, kita mengosongkan Initial Valuenya, maka jika tidak diisi variabel ini akan berisi string kosong. Sekarang timbul pertanyaan, kapan variabel Application Address ini diisi dengan nilai URL yang sesungguhnya? Jawabannya adalah pengisian variabel ini dilakukan melalui Smart Object Event sbb :

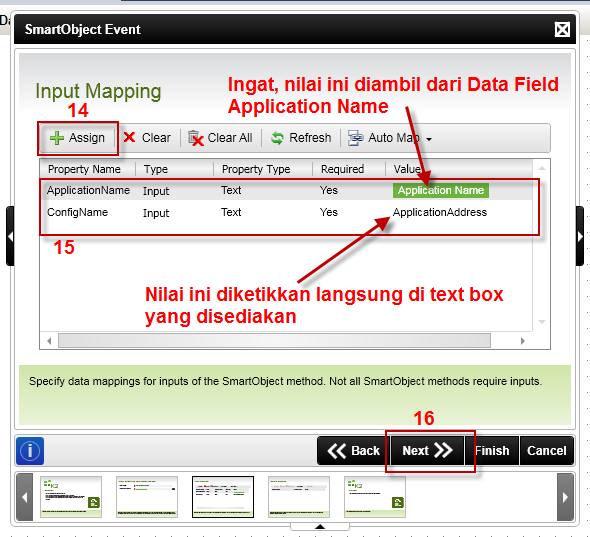
**Penting** : sebelum mempraktekkan step by step berikut ini, tambahkan dulu sebuah Data Field bernama Application Name bertipe string, dan Initial Valuenya adalah TDM sbb :

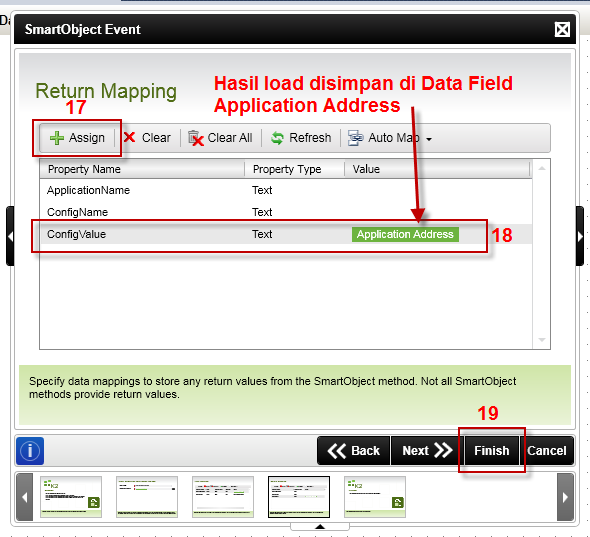


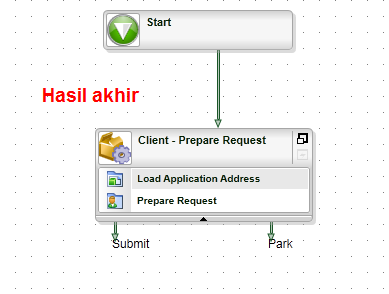




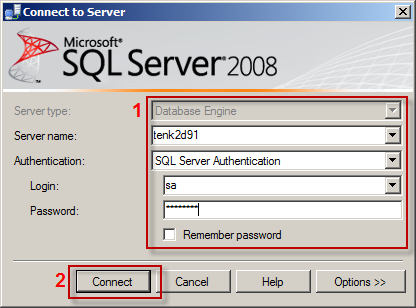


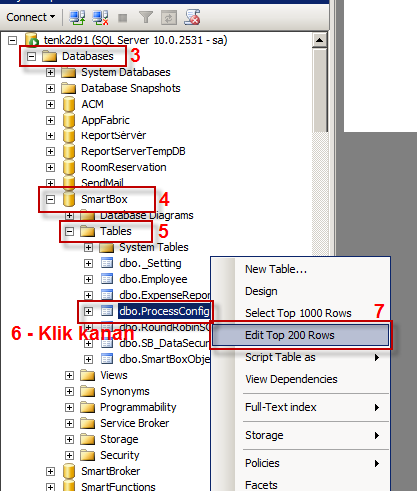


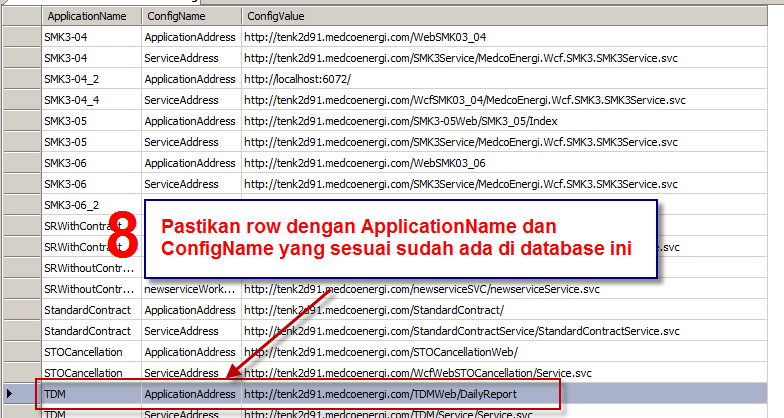




Di langkah no 15 di atas, kita mengisikan nilai dari variabel Application Name (yaitu TDM) sebagai parameter ApplicationName, dan kita mengisikan ApplicationAddress sebagai parameter ConfigName. Kita harus mengingat dua hal tersebut ketika akan mengisikan URL yang sesungguhnya dari form isian, yang dilakukan melalui SQL Server Management Studio sbb :







Dengan pengaturan yang sudah kita buat ini, Data Field Application Address akan terisi dengan nilai <http://tenk2d91.medcoenergi.com/TDMWeb/DailyReport> setiap kali smart object event Load Application Address dijalankan. Keuntungan dari pengaturan semacam ini adalah jika suatu saat ada perubahan alamat form (atau alamat2 lainnya), kita tinggal mengubah row di database tanpa harus melakukan kompilasi dan deploy ulang dari workflow.

Server Event - 1

Jenis event di k2 workflow yang perlu kita kenal adalah Server Event. Di event inilah kita bisa memasukkan coding C# yang diperlukan untuk memanipulasi workflow. Seperti yang sudah kita bahas sebelumnya, isi coding di server event ini hanyalah pemanggilan ‘otak’ dari workflow yaitu WCF service yg sudah kita buat sebelumnya, lalu menyimpan hasilnya di Data Field. That’s it. Pengaturan semacam ini dilakukan karena coding server event di K2 praktis tidak bisa di debug, sehingga kita hindari memasukkan hal yang kompleks di sana, dan sebagai gantinya hal2 yang kompleks kita lakukan di WCF service library yang lebih mudah maintenancenya.

Untuk langkah awal, kita akan membiarkan saja coding di server event kosong, dan fungsinya lebih sebagai kerangka dari fungsi2 yang akan dibuat di WCF service sebagai otaknya workflow.

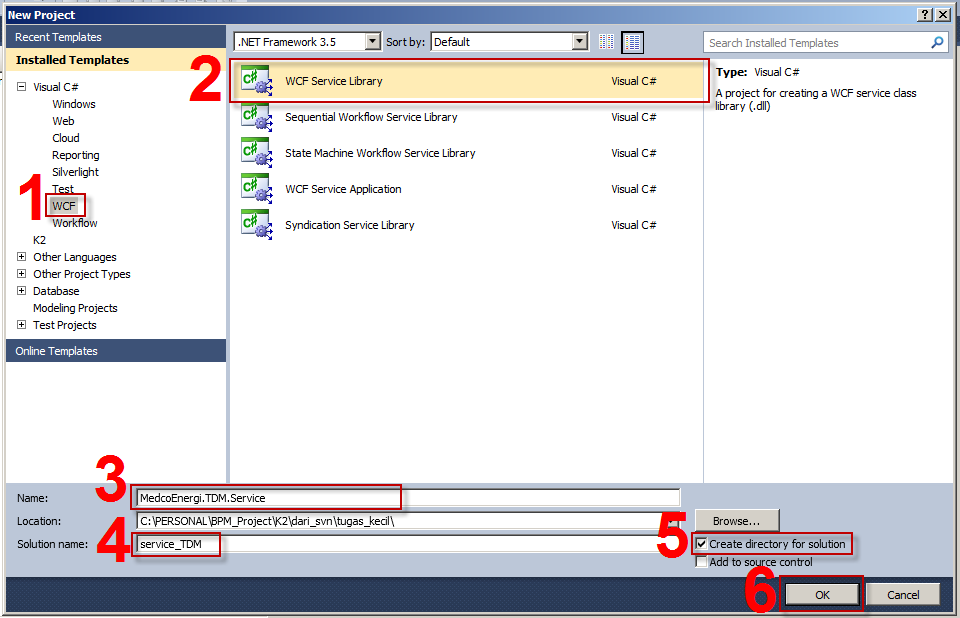
Untuk workflow TDM, kita perlu melakukan approval bertahap hingga manager L4. Untuk itu kita perlu mendapatkan posisi awal dari si requester ketika memulai TDM workflow ini. Operasi ini akan jadi satu server event, dan sekaligus menjadi satu fungsi yang perlu dibuat di WCF servicenya. Kita akan namakan fungsi itu GetRequesterPosition dengan parameter input adalah sebuah process ID dari workflow yang dimulai oleh si requester, serta outputnya adalah position ID dari si requester.

Setelah posisi awal si requester didapatkan, selanjutnya kita perlu mendapatkan atasan dari posisi tersebut, sekaligus mengecek level dari posisi tersebut. Bila seorang manager L4 sudah memberikan approval, artinya approval tidak perlu dilanjutkan ke atasannya lagi. Operasi ini lagi2 menjadi satu server event di workflow, dan sekaligus menjadi satu fungsi lagi yang perlu dibuat di WCF servicenya. Kita akan namakan fungsi itu GetNextApprover, dengan 2 parameter input yaitu current approver dan process ID, serta dua parameter output, yaitu position ID atasan dari current approver, serta sebuah status yang menunjukkan apakah approval perlu dilanjutkan atau tidak.

Dua fungsi ini bisa kita gunakan sebagai acuan untuk mulai membuat kerangka dari WCF servicenya.

Membuat WCF service sebagai otak dari workflow

Berbekal dua fungsi yang kita tahu akan diperlukan di workflow TDM tadi, kita bisa mulai membuat WCF servicenya. WCF service ini akan kita buat sebagai sebuah solution yang terpisah dari workflownya sbb :



By default, akan ada dua file dalam project WCF ini : IService.cs dan Service.cs. Hapus kedua file ini, dan sebagai gantinya, tambahkan sebuah interface baru bernama ITdmWorkflowService.cs yang berisi signature fungsi yang kita perlukan sbb:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.ServiceModel;

namespace MedcoEnergi.TDM.Service

{

[ServiceContract]

interface ITdmService

{

[OperationContract]

bool GetRequesterPosition(int processId, out string requesterPosition, out string errorMessage);

[OperationContract]

bool GetNextApprover(int processId, string currentApprover, out string nextApprover, out bool isValidNextApprover, out string errorMessage);

}

}

ITdmService.cs :

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.ServiceModel;

Tambahkan juga sebuah class untuk mengimplementasi interface tersebut yaitu TdmWorkflowService.cs sbb:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace MedcoEnergi.TDM.Service

{

public class TdmWorkflowService : ITdmWorkflowService

{

public bool GetRequesterPosition(int processId, out string requesterPosition, out string errorMessage)

{

throw new NotImplementedException();

}

public bool GetNextApprover(int processId, string currentApprover, out string nextApprover, out bool isValidNextApprover, out string errorMessage)

{

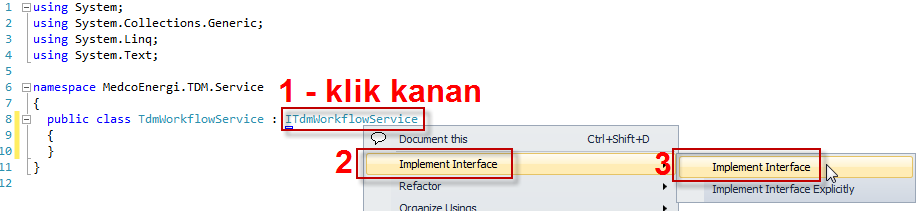
throw new NotImplementedException();

}

}

}

Tips : Untuk mendapatkan file seperti tsb di atas dengan mudah, lakukan hal berikut :



Seperti kita lihat, implementasi dari service ini masih default, yaitu throw exception. Untuk sekarang, kita biarkan dulu seperti itu karena yang kita perlukan saat ini hanyalah signature dari fungsi tersebut yang ada di interfacenya.

Langkah terakhir sebelum service ini bisa di-publish adalah mengubah file konfigurasi app.config sbb (cukup ubah bagian yang berwarna kuning dengan nama class dan interface yang ingin kita publish) :

....

<services>

<service name="MedcoEnergi.TDM.Service.TdmWorkflowService" behaviorConfiguration="MedcoEnergi.TDM.Service.Service1Behavior">

<host>

<baseAddresses>

<add baseAddress = "http://localhost:8732/Design\_Time\_Addresses/MedcoEnergi.TDM.Service/Service1/" />

</baseAddresses>

</host>

<!-- Service Endpoints -->

<!-- Unless fully qualified, address is relative to base address supplied above -->

<endpoint address ="" binding="wsHttpBinding" contract="MedcoEnergi.TDM.Service.ITdmWorkflowService">

<!--

Upon deployment, the following identity element should be removed or replaced to reflect the

identity under which the deployed service runs. If removed, WCF will infer an appropriate identity

automatically.

-->

<identity>

<dns value="localhost"/>

</identity>

</endpoint>

<!-- Metadata Endpoints -->

<!-- The Metadata Exchange endpoint is used by the service to describe itself to clients. -->

<!-- This endpoint does not use a secure binding and should be secured or removed before deployment -->

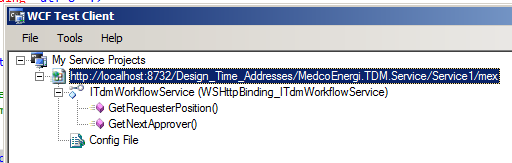
<endpoint address="mex" binding="mexHttpBinding" contract="IMetadataExchange"/>

</service>

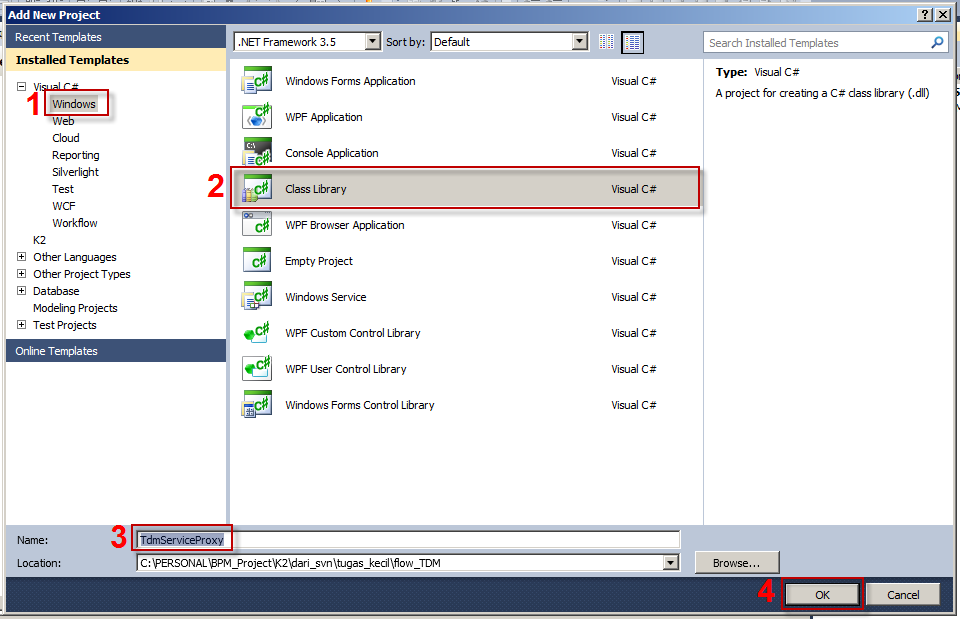
</services>

....

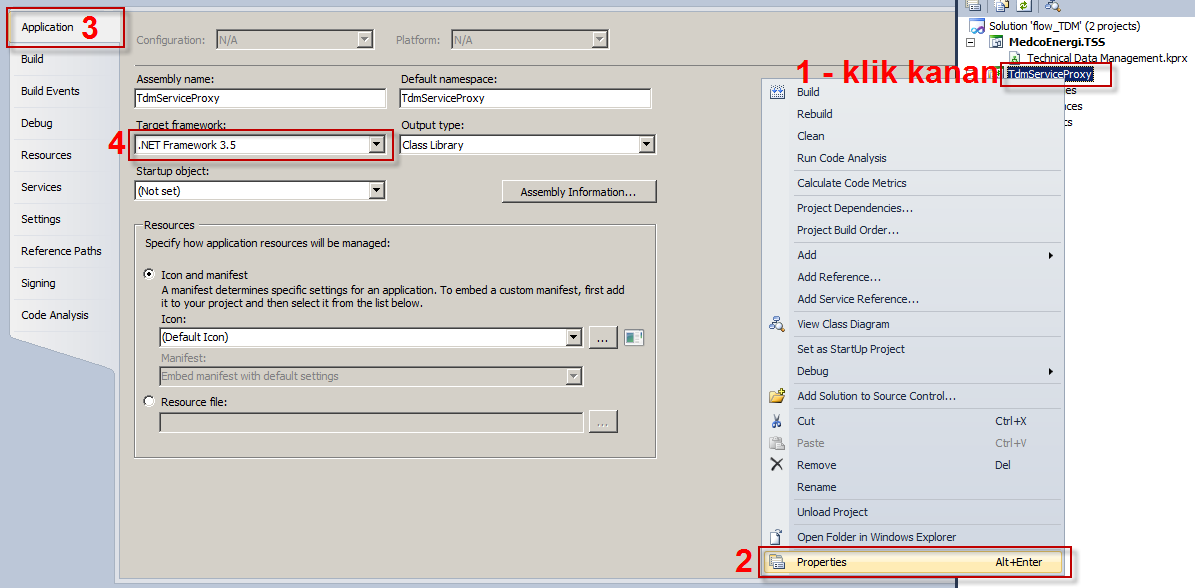
Setelah itu, silahkan tekan F5 dari Visual Studio untuk menjalankan service yang sudah kita buat :



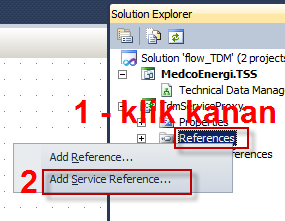
Dengan kondisi service tersebut masih terpublish, silahkan kembali ke project K2 workflow yang kita buat pertama kali. Tambahkan sebuah project baru bertipe class library di solution tersebut. Class library ini akan kita gunakan untuk mengakses WCF service yang sudah kita publish tadi.

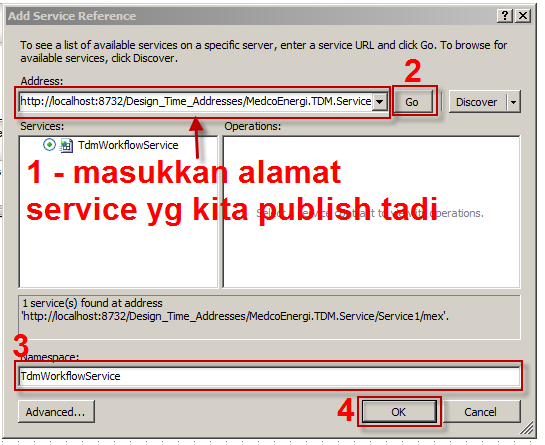


Pastikan bahwa .NET framework yang digunakan untuk class library ini adalah versi 3.5 :



Hapus file Class1.cs yang ditambahkan secara default, karena kita tidak membutuhkannya. Tambahkan service reference ke class library tsb sbb :





Setelah selesai, kita akan menambahkan sebuah class baru untuk memudahkan kita mengakses service tersebut. Yang dilakukan class ini sebenarnya hanyalah menyediakan binding ke service tersebut secara programmatically, sehingga dari workflow nanti kita hanya perlu menyediakan addressnya saja. Beri nama class tersebut WcfClientFactory.cs, isinya sbb :

namespace TdmServiceProxy

{

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using TdmService;

using System.ServiceModel;

public static class WcfClientFactory

{

public static TdmWorkflowServiceClient CreateTdmServiceClient(string serviceAddress)

{

WSHttpBinding httpBinding = new WSHttpBinding();

return new TdmWorkflowServiceClient(httpBinding, new EndpointAddress(serviceAddress));

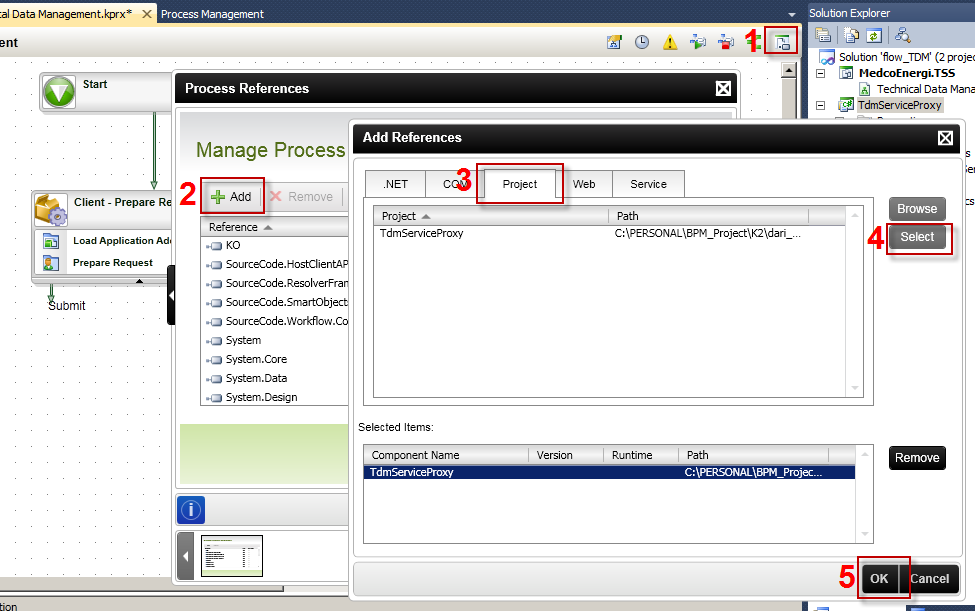
}

}

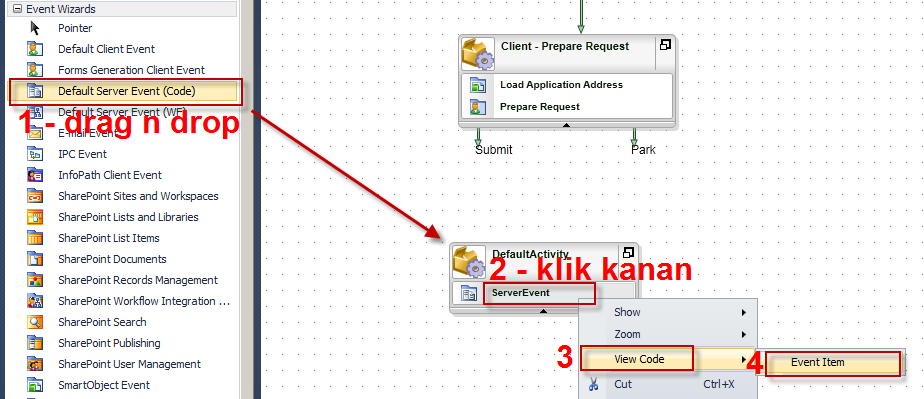
}

Yang perlu diperhatikan, pastikan class tersebut bertipe static, dan binding yang digunakan harus sesuai dengan binding yang digunakan pada WCF service yang kita buat tadi. Dalam contoh ini kita menggunakan WS Http Binding.

Langkah selanjutnya adalah menambahkan project tersebut sebagai reference pada project K2 workflow kita sbb :



Selanjutnya tambahkan server event pertama untuk mendapatkan posisi dari requester sbb :



Pada coding C# yang muncul, tambahkan kode berikut untuk mendapatkan requester position melalui WCF service yang sudah kita buat :

var client = WcfClientFactory.CreateTdmServiceClient(K2.ProcessInstance.DataFields["ServiceAddress"].Value.ToString());

try

{

string requesterPosition;

string errorMessage;

if (!client.GetRequesterPosition(out requesterPosition, out errorMessage, K2.ProcessInstance.ID))

{

throw new Exception(errorMessage);

}

K2.ProcessInstance.DataFields["RequesterPosition"].Value = requesterPosition;

K2.ProcessInstance.DataFields["Participant"].Value = requesterPosition;

}

finally

{

client.Close();

}

Jangan lupa juga tambahkan SmartObject Event tepat sebelum server event tsb di atas untuk me-load alamat service yang disimpan pada data field ServiceAddress. Caranya silahkan refer pada contoh sebelumnya ketika me-load application address.