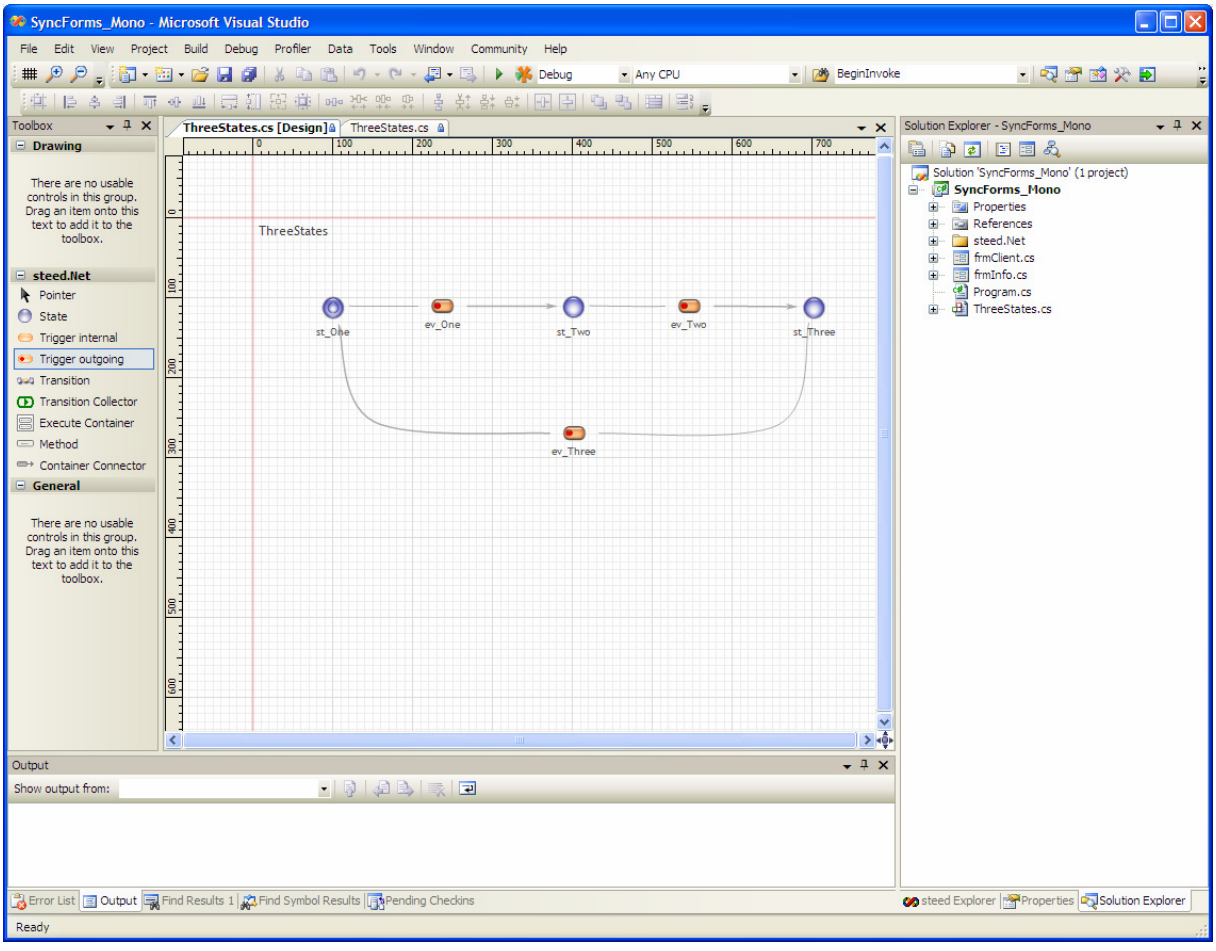
Sample 2 - C/S nur steed.Net Enterprise

1. Legen Sie eine leere Solution an.
2. Fügen Sie dieser eine Console Apllikation hinzu(Server)
3. Erstellen Sie eine neue Zustandsmaschine
   Hinzufügen - > neues Element -> steed.net -> Statemachine.
   Benennen Sie diese Datei als ThreeStates.cs
4. Wechseln Sie in den Designer von ThreeStates
5. Statemachine so Designen das sie so aussieht
   a. fügen Sie 2 States hinzu (einer existiert schon)
   b. Vergeben Sie DisplayNames wie folgt : St_One, St_Two, St_Three
   c. Verbinden Sie St_One und St_Two mit einer neuen Transition, setzen Sie den DisplayName des Triggers dieser Transition auf „ev_One“ und den TriggerType auf Incoming. Von ST_One nach St_Two !
   d. Verbinden Sie St_Two und St_Three mit einer neuen Transition, setzen Sie den DisplayName des Triggers dieser Transition auf „ev_Two“ und den TriggerType auf Incoming. Von ST_Two nach St_Three !
   e. Verbinden Sie St_Three und St_One mit einer neuen Transition, setzen Sie den DisplayName des Triggers dieser Transition auf „ev_Three“ und den TriggerType auf Incoming. Von ST_Three nach St_One !
   f. Stellen Sie sicher das St_One der initialstate ist.



6. Nun kommt das wichtigste dieses Projektes.
   - Jeder Client wird später eine Referenz auf eine Zustandsmaschine „ThreeStates“ besitzen
   - Im normal Fall hat dieses Szenario zu Folge das jede Instanz vom Client eine Anfrage an den Router stellt eine neue Instanz zu erstellen.
   - Das würde bedeuten das jeder Client mit seiner „eigenen“ Instanz von ThreeStates arbeiten würde und der Effekt der Synchronisation ausbliebe.

7. Öffnen Sie das Code-Fenster der Zustandsmaschine ThreeStates
   Sie sehen den Guid-Attribute oberhalb des class Schlüsselwortes.
   Fügen Sie der Klasse einen weiteren Attribut hinzu wie folgt :
   [HOBSingleton(true)]
   Dieser Attribut bewirkt das der erste Anfragende einer Instanz mit einer neuen Instanz bedient wird, jedoch jeder weitere Anfragende die gleiche Instanz geliefert bekommt.

```
/// </summary>

//This Attribute sets an unique Identifier to th
[Guid("36346aff-ced6-4cf8-aec1-73e51ba8565f")]
[HOBSingleton(true)]
public class ThreeStates : Statemethod.Statemach
{
    public static readonly Guid _GUID = new Guid
    private const string _sGUID = "36346aff-ced6-
    private State state2;
    private State state3;
    private Transition transition1;
    private Trigger trigger1;
```

8. Ergänzen Sie die Main-Prozedur folgendermaßen

```
static void Main(string[] args)
{
   Statemethod.Router.Router.Start(true);
   Console.ReadLine();
   Statemethod.Router.Router.Stop(true);
}
```

9. Fügen Sie diesem Projekt eine Router-Konfiguration hinzu.
   Hinzufügen - > neues Element -> steed.net -> Application.config

Nähere Beschreibung einzelner Konfigurations-Parameter finden Sie in der Konfigurationsdatei.

10. Erforderliche Parameter für diese Router-Konfiguration sind:
    a. `AutoScanForRouters = "true"`
    b. `ReScanForRoutersTime = "5"`
    c.
```
<ComProviders>
<Provider
      Enabled          = "true"
      Assembly    = "Statemethod.ComProvider.WKObject,
Version=2.0.0.8, Culture=neutral, PublicKeyToken=6143c6c52ea203af"
      Type        =
"Statemethod.ComProvider.WKObject.WKObjectComProvider"
      ProviderID  = "41bbbc51-01e6-4051-9b2e-5cef9889e11c"
      ProviderName="SyncForms_Server.WKOBJECT" >

<Permission Enabled="true" Group="*" Permission="FullControl"
/>
            <Parameter key="TCPPortRange" value="9211-9220" />
            <Parameter key="UDPPortRange" value="9171-9180" />

</Provider> …
```

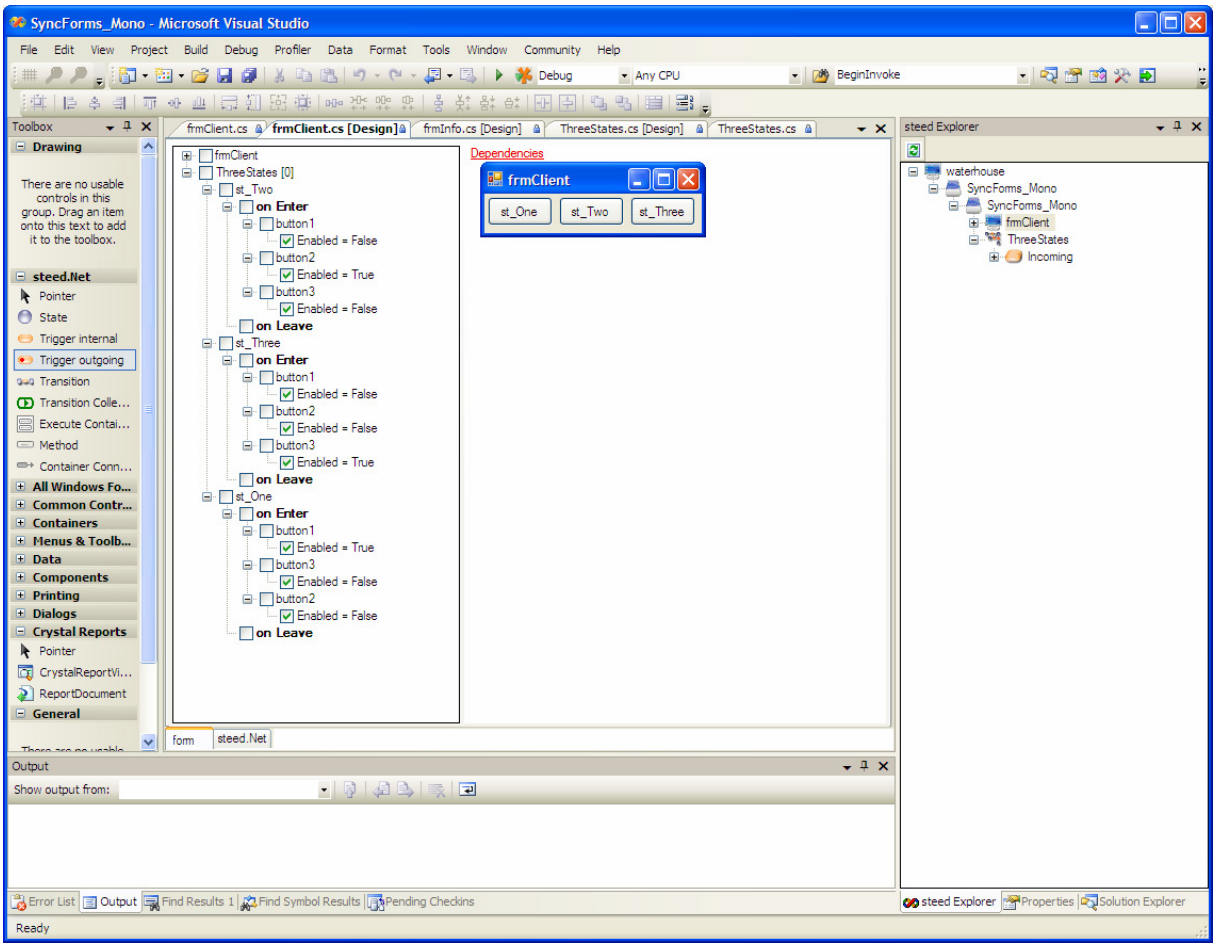
11. Mit diesen Einstellung wird sicher gestellt das diese Applikation automatisch alle 5 Sekunden nach neuen Routern sucht und auch einen Com-Provider besitzt.
12. Fügen Sie bitte nun der Solution ein neues Projekt vom Typ WindowsApplication hinzu.
13. Fügen Sie dem Projekt eine StatemachineForm (frmClient) hinzu. Hinzufügen - > neues Element -> steed.net -> StatemachineForm.
14. Öffnen Sie den Designer dieser Form und wechseln dort in den „steed.net“ Designer (Tab unten links)
15. Wechseln Sie nun in das Fenster „steed-Explorer“ und suchen Sie die zuvor erstellte Zustandsmaschine „ThreeStates“ aus dem Projekt Server
    Sollten Sie dieses Fenster nicht sehen so finden Sie dieses unter dem Menüpunkt „steed Explorer“ im Ansicht Menü.
    a. Ziehen Sie nun ThreeStates aus dem steed-Explorer in den steed-Designer Ihrer frmClient.
    b. Sie haben nun 2 Zustandsmaschinen mit einander verbunden indem Sie eine StatemachineReference erzeugt haben.

16. Wechseln Sie in den Form-Designer (Tab unten links)
    a. Erstellen Sie auf dieser drei buttons.
       button1.Text = „st_One“
       button2.Text = „st_Two“
       button2.Text = „st_Three“
       Fügen Sie jedem Button ein Click-Event hinzu und ergänzen Sie folgenden Code

```
private void button1_Click(object sender, EventArgs e)
    {
       this.statemachineReference1.SendMessage("ev_One");
    }

    private void button2_Click(object sender, EventArgs e)
    {
       this.statemachineReference1.SendMessage("ev_Two");

    }

    private void button3_Click(object sender, EventArgs e)
    {
       this.statemachineReference1.SendMessage("ev_Three");

    }
```

    b. Bitte achten Sie auf Groß-/Kleinschreibung der Displaynamen

17. Wechseln Sie nun wieder in die Ansicht des Windows-Form-Designers
    a. Oberhalb der Form sehen Sie einen Link „Dependencies“. Bitte klicken Sie dort.



Dependencies sind Objekte die in der Lage sind Eigenschaften eines Objektes anhand einer Zustandsänderung einer Zustandsmaschine zu verändern.

18. Um ein Dependency zu erstellen suchen Sie sich in dem erschienen TreeView die Zustandsmaschine heraus auf dessen Zustandsveränderung Sie reagieren wollen.

19. Öffnen Sie die Zustandsmaschine „ThreeStates“
    a) Sie sehen nun alle verfügbaren Zustände dieser Zustandsmaschine
    b) Öffnen Sie nun „st_One“
    c) Es werden Ihnen nun zwei weitere Einträge gezeigt.
    d) „on Enter“ wird ausgelöst sobald ein Zustand erreicht wird.
    e) „on Leave“ wird ausgelöst wenn ein Zustand verlassen wird.
    f) Markieren Sie nun „on Enter“ von „st_One“ der Zustandsmaschine „ThreeStates“
    g) Markieren Sie nun das Control welches in diesem Zustand eine Veränderung erfahren soll.
    h) Markieren Sie den Button mit dem Text „st_One“ (button1)
    i) Nun müssen Sie nur noch bestimmen welche Änderungen im Zustand „st_One“ auf den Button „st_One“ (button1) angewendet werden sollen.
    j) Ändern Sie nun im PropertyGrid die Eigenschaft Enabled auf True. Da dieser Button schon die Eigenschaft Enabled auf True besitzt wechseln Sie bitte auf False und dann wieder auf True

Der Dependency-Designer reagiert nur auf Eigenschaftsveränderungen. So kann es sein das Sie erst eine Eigenschafts-Veränderung machen müssen und diese wieder zurücknehmen müssen damit der Designer den gewünschten Wert erkennt.

Die Dependencies werden nur erkannt wenn :
„on Enter“ oder „on Leave“ eines Zustandes markiert ist.
ein Control bzw. Element im Designer markiert ist.
Eine Änderung im Propertygrid vorgenommen wird.

20. Sobald Sie eine Änderung im Propertygrid gemacht haben erzeugt der Dependency-Designer einen neunen Eintrag im TreeView unter „on Enter“.
21. Mit der Checkbox vor jedem Eintrag im TreeView können Sie definierte Dependencies an- oder ausschalten ohne Sie löschen zu müssen.

22. Bitte designen Sie für jeden State und jedes Control folgendes :

„st_ One“ – „on Enter“ – button1.enabled=true
„st_ One“ – „on Enter“ – button2.enabled=false
„st_ One“ – „on Enter“ – button3.enabled=false
„st_ Two“ – „on Enter“ – button1.enabled=false
„st_ Two“ – „on Enter“ – button2.enabled=true
„st_ Two“ – „on Enter“ – button3.enabled=false
„st_ Three“ – „on Enter“ – button1.enabled=false
„st_ Three“ – „on Enter“ – button2.enabled=false
„st_ Three“ – „on Enter“ – button3.enabled=true

23. Bitte speichern Sie Ihr Projekt.
24. Fügen Sie nun Ihrem Projekt eine neue Windows-Form hinzu und benennen Sie dies als „frmMain“.

25. Erzeugen Sie auf dieser einen Button mit dem Text „create Client“
    Im Click-Handler dieses Buttons ergänzen Sie folgenden Code :
    Den button ClickHandler – fogendes

```
private void button1_Click(object sender, EventArgs e)
    {
        frmClient frm = new frmClient();
        frm.Show();
    }
```

26. Erstellen Sie in dieser eine Main-Methode die wie folgt aussieht :

```
[STAThread]
    static void Main()
    {
        Application.EnableVisualStyles();
        Application.SetCompatibleTextRenderingDefault(false);
        Statemethod.Router.Router.Start(true);
        Application.Run(new frmMain());
        Statemethod.Router.Router.Stop(true);
    }
```

27. Fügen Sie diesem Projekt eine Router-Konfiguration hinzu.
    Hinzufügen - > neues Element -> steed.net -> Application.config

> Nähere Beschreibung einzelner Konfigurations-Parameter finden Sie in der Konfigurationsdatei.

28. Erforderliche Parameter für diese Router-konfiguration sind:
    a. `AutoScanForRouters = "true"`
    b. `ReScanForRoutersTime = "5"`
    c.

```
<ComProviders>
    <Provider
          Enabled          = "true"
          Assembly    = "Statemethod.ComProvider.WKObject,
Version=2.0.0.8, Culture=neutral, PublicKeyToken=6143c6c52ea203af"
          Type        =
"Statemethod.ComProvider.WKObject.WKObjectComProvider"
          ProviderID  = "41bbbc51-01e6-4051-9b2e-5cef9889e11c"
          ProviderName="SyncForms_Server.WKOBJECT" >

    <Permission Enabled="true" Group="*" Permission="FullControl"
/>
                <Parameter key="TCPPortRange" value="9211-9220" />
                <Parameter key="UDPPortRange" value="9171-9180" />

    </Provider> …
```

29. Mit diesen Einstellung wird sicher gestellt das diese Applikation automatisch alle 5 Sekunden nach neuen Routern sucht und auch einen Com-Provider besitzt.

30. Speichern sie die Solution , starten Sie den Server (ConsolenApplikation) und danach den Client (WindowsApplikation)