

Überblick über die Möglichkeiten und Grenzen der NX-Zusatzprogrammierung

(Macro, Journal, SNAP, NXOpen, GRIP, KF)

PLM Benutzergruppe
Seeheim / 04.07.2017

HBB Engineering GmbH
Referent: Walter Hogger / Andreas Seiwald
Zielgruppe: NX-Administratoren

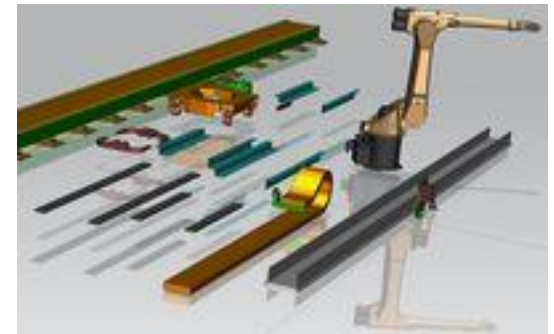
Zur Firma

- Die **HBB Engineering GmbH** wurde am 24.04.1999 gegründet.
- Die Firma ging aus dem "Ingenieurbüro Walter Hogger" hervor.
Walter Hogger arbeitet **seit 1985 (über 32 Jahre)** im Umfeld von **NX** (früher UNIGRAPHICS).
- Die HBB Engineering GmbH beschäftigt derzeit **18** fest angestellte **Mitarbeiter**.
- **NX-Themen:** NX-Verkauf, NX-Installation, NX-Training, NX-Programmierung, NX-Dokumentation, NX-Bücher, NX-Hotline, NX-Forum, Konstruktion (mit NX und Catia V5)
- NX-Anwender & Dienstleister
- **Standorte:** Anger/Bayern und Ruhla/Thüringen



Was versteht man unter NX-Programmierung?

- 1993 UNIGRAPHICS V10, neues Konzept mit parametrischen Features, Programmierfähigkeit nimmt ab ... derzeit nimmt sie wieder stark zu.
- Automatisierung von Abläufen oder Vorgängen in Siemens NX
- NX-Funktionen in eigene Programme packen:
 - Generieren von nicht vorhandenen Funktionen
 - Firmen-Know-how von in Programmform „Gießen“
 - Massendaten abarbeiten ([Beispiel](#))
 - Intelligente Baukastenlösungen ([Beispiel](#))
 - Komplizierte Vorgänge beschleunigen ([Beispiel](#))
 - Erstellung von vereinfachten, leicht verständlichen Dialogen
 - Behebung von bestehenden Programmfehlern



Überblick über NX-Programmierung / Automatisierung

	Beschreibung	Sprache
Macro	Nicht editierbar, Unstabil, schlecht aufwärtskompatibel	-
Journaling	Unvollständige Aufzeichnung, gut editierbar, läuft ohne Kompilierung	C#, VB.NET, Python (C++, Java)
GRIP	Veraltet, nur in NX verfügbar, Befehlsumfang begrenzt	GRIP
NXOpen	Objektorientierte Bibliothek, sehr umfangreich, moderne Programmiersprachen	C#, VB.NET, Python, C, C++, Java
User Function	Funktionsbasierte Bibliothek, sehr umfangreich, moderne Programmiersprachen	C#, VB.NET, Python, C, C++, Java
SNAP	Vereinfachte Bibliothek, nur für .NET-Sprachen, „GRIP-Ersatz“	C#, VB.NET,
Knowledge Fusion	verschiedene Möglichkeiten Intelligenz mit Parts, Geometrie zu verknüpfen, aufwendig	KF

Entwickler-Lizenzen für die NX-Programmierung

Lizenz	Beschreibung	Sprache
Ohne Lizenz	Journaling	C#, VB.NET, Python
NX Open for .NET Author	Erstellen von kompilierten .NET-Programmen (NXOpen, UF)	C#, VB.Net
NX Open Toolkits Author	Erstellen von NX-Programmen in den Programmiersprachen	C, C++, Java
NX Open Dialog Designers	Erstellen von NX-ähnlichen Dialogen	C#, VB.NET, Python, C, C++, Java
NX SNAP Author	Erstellen von SNAP-Programmen	C#, VB.Net
NX KF Author	Erstellen von KF-Programmen	KF

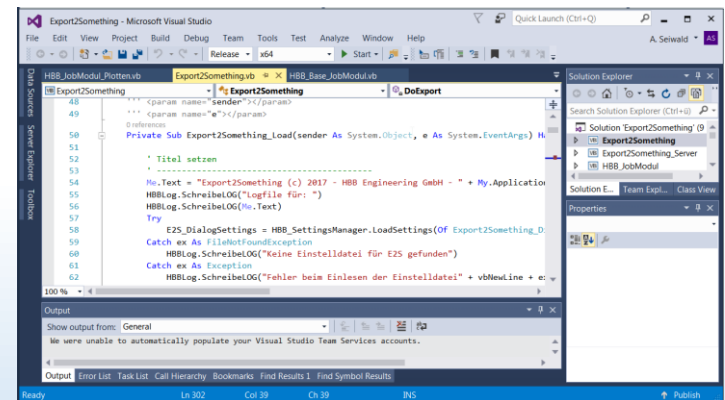
Signieren / Unterschreiben von Programmen

- Ausführung nicht signierter Programme erfordert verfügbare Autorenlizenz!
- Programme können zur Weitergabe mit Autorenlizenz unterschrieben werden!
- Für die Signierung ist die Einbettung der „NXSigningResource.res“ in das Programm notwendig (siehe UGOPEN-Verzeichnis).
- Um zu signieren wird das Hilfsprogramm „SignDotNet.exe“ aufgerufen
NX 8.5 – NX10.0: %UGII_BASE_DIR%\UGII\SignDotNet.exe
NX 11.0: %UGII_BASE_DIR%\NXBIN\SignDotNet.exe
- SNAP-Programme erfordern einen zusätzlichen Parameter (-snap oder -both)

```
C:\Siemens\NX11.0\NXBIN>SignDotNet.exe
Usage: SignLibrary [<switches>] <filenames...>
<switches> are optional and may be one of the following:
  -help      prints this message
  -verify    verifies that the files have been signed correctly
  -snap      sign with SNAP signature
  -both      sign with both SNAP and NXOpen signature
NOTE: the -snap and -both option cannot be used at the same time.
```

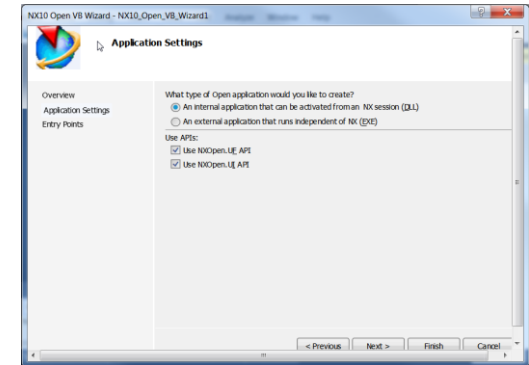
Software-Voraussetzung für .NET-Programmierung

- Rechner mit Windows-Installation (7, 8, 10)
- Vollständige NX-Installation (Programmierschnittstellen)
- Entwicklungsumgebung für .NET-Programmiersprachen
 - Microsoft Visual Studio (evtl. Community-Edition)
 - SharpDevelop (OpenSource Entwicklungsumgebung)
- Siemens empfiehlt folgende Visual Studio Versionen
 - NX 9.0: Microsoft Visual Studio 2012
 - NX 10.0: Microsoft Visual Studio 2012
 - NX 11.0: Microsoft Visual Studio 2013



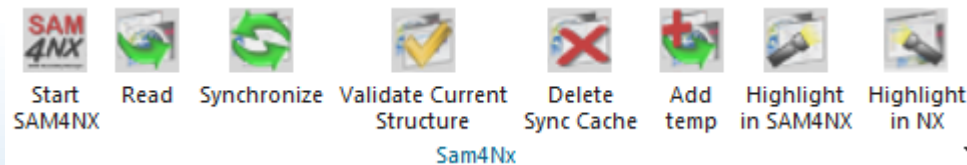
Visual Studio Vorlagen für NX-Programme

- Visual Studio Wizard für NXOpen-Programme
 - %UGII_BASE_DIR%\UGOPEN\vs_files
 - Inhalt kopieren nach
C:\Program Files (x86)\Microsoft Visual Studio xx.0
- Visual Studio Wizard für SNAP-Programme
 - %UGII_BASE_DIR%\UGOPEN\SNAP\Templates
 - Kopieren in des eingestellte Template-Verzeichnis von Visual Studio
- NX-Programme können auch ohne Wizards erstellt werden!

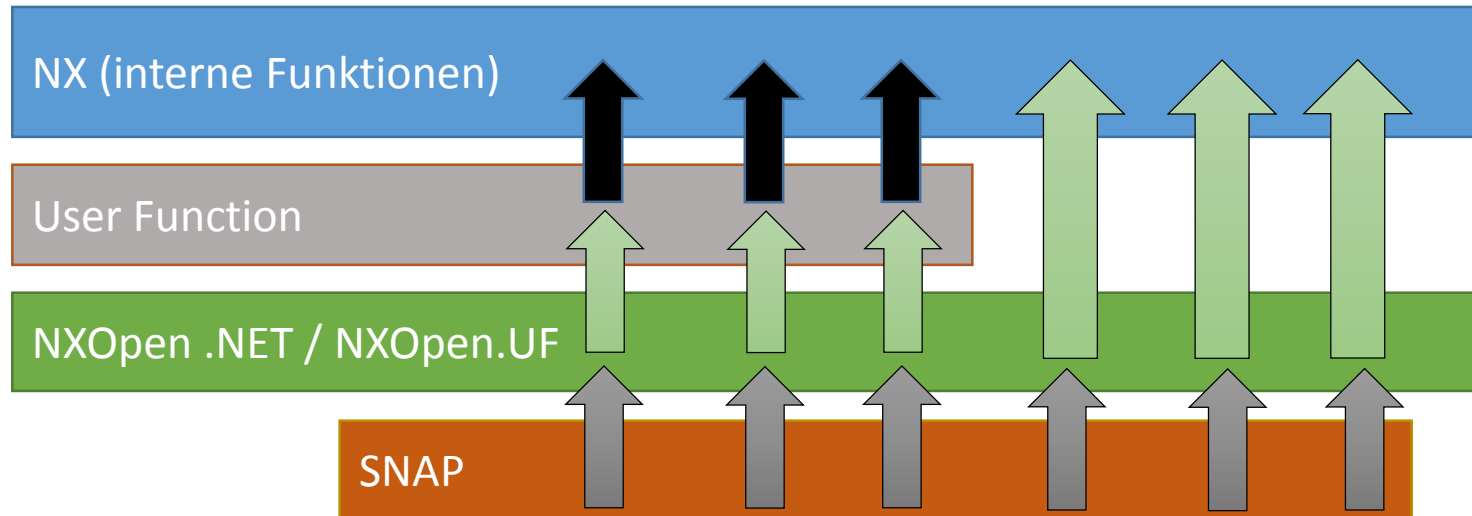


Es können verschiedene Programmtypen erzeugt werden!

- Internes NX-Programm (*.dll)
 - Innerhalb eines laufenden NX-Prozesses wird ein Programm ausgeführt, z.B. durch einen Klick auf ein Icon
- Externes NX-Programm (*.exe)
 - Ein externes Programm wird direkt in Windows gestartet. Die NX-Oberfläche ist nicht sichtbar, aber eine NX-Installation muss vorhanden sein.
- Remoting-Zusatzprogramm (*.dll + *.exe)
 - Es gibt ein Server (*.dll) und Client-Programm (*.exe). Damit dann ein laufender und sichtbarer NX-Dialog quasi von außen ferngesteuert werden.
- Daimler SAM4NX

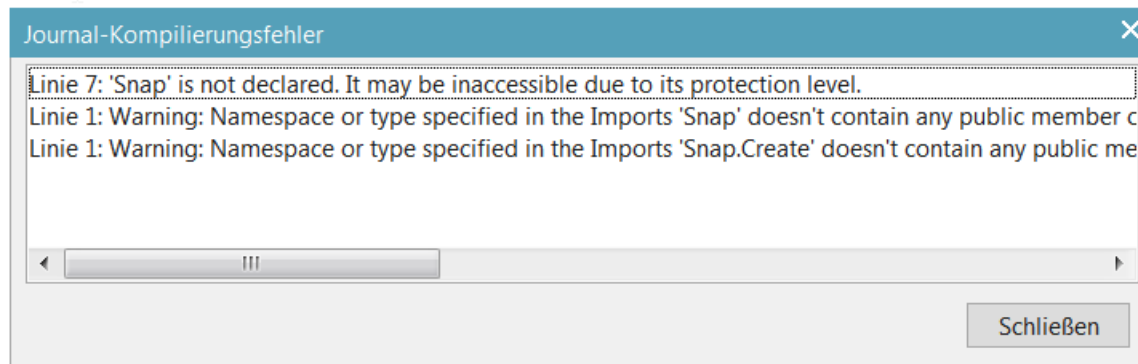


Aufbau / Umfang der Bibliotheken



Journaling mit NXOpen und SNAP

- Über Journaling kann auf die Bibliotheken NXOpen und UF (User Function) zugegriffen werden!
- SNAP lässt kein Journaling zu! (SNAP Namensraum ist nicht verfügbar...)



NXOpen (.NET)

- Objektorientierte Bibliothek
- Wird in Journalen verwendet
- Hauptsächlich neuere NX-Befehle enthalten
- einfache Funktionen verursachen lange Programmcodes, siehe z.B. zum Erstellen einer assoziativen Linie („Builder“)

```
' Part Objekt
Dim Prt As Part = theSession.Parts.Display

' Schleife über alle Kurven im Part-Objekt
For Each Kurve As Curve In Prt.Curves
    Kurve.Layer = 30          ' Layer ändern
Next
```

```
Dim nullFeature As Features.AssociativeLine = Nothing
Dim Startpunkt As Point = theSession.Parts.Work.Points.CreatePoint(New Point3d(30, 30, 30))
Dim EndPunkt As Point = theSession.Parts.Work.Points.CreatePoint(New Point3d(50, 50, 50))

Dim LineBuilder1 As Features.AssociativeLineBuilder =

theSession.Parts.Work.BaseFeatures.CreateAssociativeLineBuilder(nullFeature)

LineBuilder1.StartPointOptions = Features.AssociativeLineBuilder.StartOption.Point
LineBuilder1.EndPointOptions = Features.AssociativeLineBuilder.EndOption.Point

LineBuilder1.StartPoint.Value = Startpunkt
LineBuilder1.EndPoint.Value = EndPunkt

Dim NxObj_Line As NXObject = LineBuilder1.Commit
LineBuilder1.Destroy()
```

NXOpen.UF

- Sehr umfangreiche funktionsbasierte Bibliothek
- Funktionen arbeiten mit sogenannten „Tags“ und Basisdatentypen
- Funktionen sind übernommen aus der C/C++ Userfunction-Bibliothek

```
' CurveTag  
Dim CurveTag As Tag = getCurveTag()  
  
' Layer ändern  
theUfSession.Obj.SetLayer(CurveTag, 30)
```

```
' Informationen für Linie definieren  
Dim UFLineInfo As UFCurve.Line  
UFLineInfo.start_point = {0, 0, 0}  
UFLineInfo.end_point = {10, 10, 10}  
Dim createdLineTag As NXOpen.Tag = Tag.Null  
  
' nicht assoziative Linie erzeugen  
theUfSession.Curve.CreateLine(UFLineInfo, createdLineTag)  
  
' Linie auf Layer verschieben  
theUfSession.Obj.SetLayer(createdLineTag, 10)  
  
' Typnummer abfragen  
Dim Typ, Subtyp As Integer  
theUfSession.Obj.AskTypeAndSubtype(createdLineTag, Typ, Subtyp)
```

SNAP

- Objektorientierte vereinfachte Bibliothek
- Kurze Befehlsaufrufe und wenig Übergabeparameter

```
' Position angeben  
Dim Ursprung As New Position(0, 0, 0)  
  
' Block-Feature erzeugen  
Dim Block1 As NX.Block = Create.Block(Ursprung, 50, 50, 10)  
  
' Layer ändern  
Block1.Layer = 30
```

Kompatibilität von NXOpen, UF und SNAP

- Gleichzeitig verwendbar innerhalb eines Programms
- Objekte, Daten können relativ leicht ausgetauscht werden
- Vorsicht bei Vermischung, da Datentypen, Objekt ähnlich benannt sind. Es können Probleme beim Einbinden von Namensräumen entstehen!

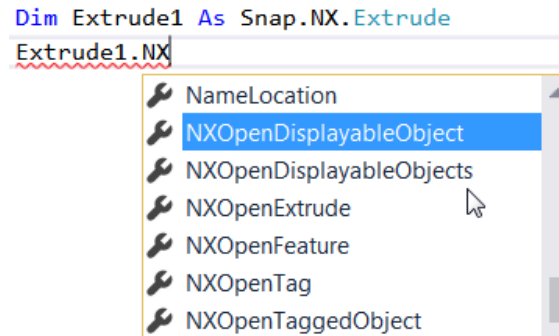
Datentyp NXOpen	Datentyp SNAP
NXOpen.NXObject	Snap.NX.NXObject
NXOpen.Line	Snap.NX.Line
NXOpen.Features.Feature	Snap.NX.Feature
NXOpen.Part	Snap.NX.Part
NXOpen.View	Snap.NX.View
NXOpen.DisplayableObject	-

Kompatibilität von NXOpen, UF und SNAP

- Sehr viele SNAP-Datentypen erlauben eine direkte Zuweisung

```
Dim L1 As Snap.NX.Line = CreateLine_SNAP()
Dim L1_NXOpen As NXOpen.Line = L1
Dim L2 As Snap.NX.Line = L1_NXOpen
```

- Zugriff auf NXOpen-Datentypen und NXOpen-Tags



- NXOpen-Tags werden in SNAP-Objekte verpackt

```
' UF-Funktionen arbeiten mit Tags
Dim Body_UF As NXOpen.Tag = GetBodyTag()
Dim Body_SNAP As Snap.NX.Body = Snap.NX.Body.Wrap(Body_UF)
```


SNAP als GRIP-Ersatz

- SNAP ist als GRIP-Ersatz von Siemens entwickelt worden
- SNAP reicht im Detail noch nicht an den Funktionsumfang von GRIP heran. Es fehlen zum Teil vergleichbare Funktionen.

- GRIP-Beispiel: Linie über Punkt und Winkel erstellen

```
L1 = LINE/Punkt1, ATANGL, 45
```

- SNAP-Beispiel: Linie über Punkt und Winkel erstellen

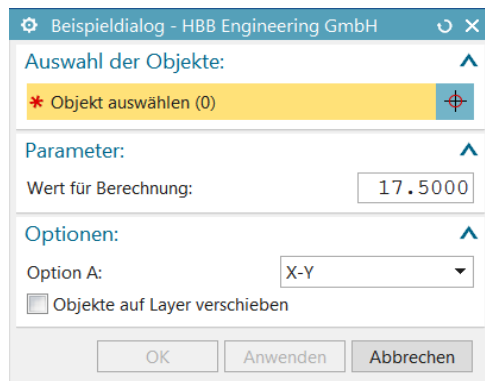
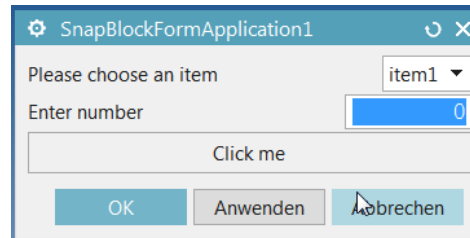
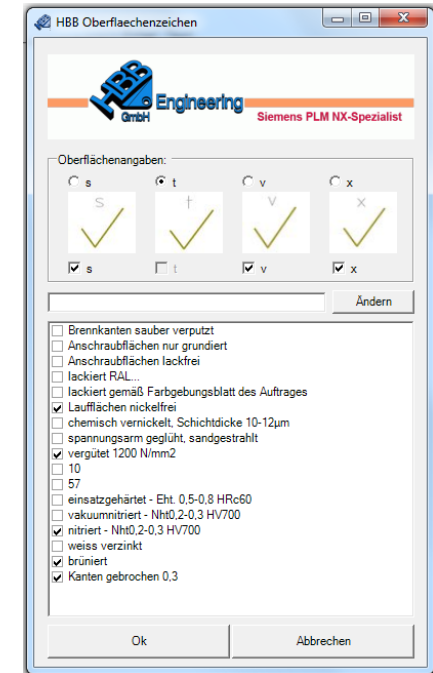
```
' SNAP erzeugt eine Linie über zwei Positionen oder Punkte
Dim Pos1 As New Snap.Position(0, 0, 0)
Dim Pos2 As New Snap.Position(10, 10, 0)
Snap.Create.Line(Pos1, Pos2)
```

- SNAP-Beispiel: Komponente zur Baugruppe hinzufügen

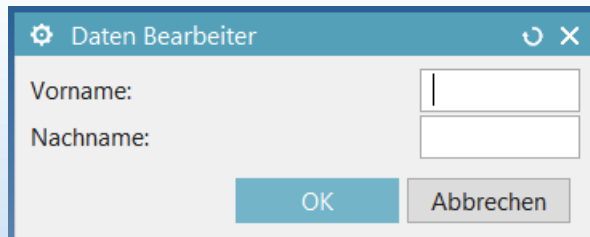
```
' Nicht möglich!
' Alternative über NXOpen ...
Snap.Globals.DisplayPart.NXOpenPart.ComponentAssembly.AddComponent(...)
```

Grafische Dialoge mit NXOpen und SNAP

- Windows Forms
- Block UI Styler und „Block Based Dialogs“

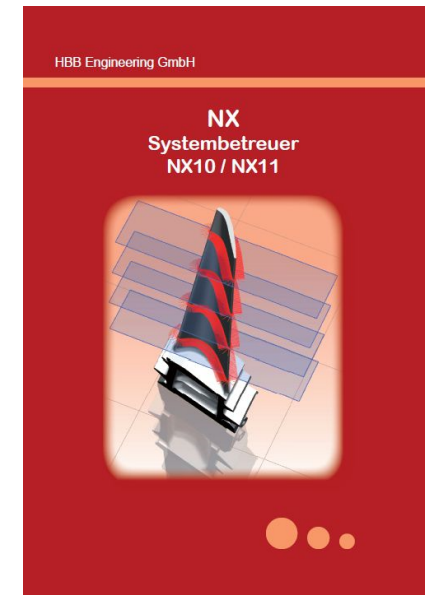




- SNAP bietet auch einfache Eingabe-Dialoge



Informationsquellen zum Thema NX-Programmierung

- Buch von HBB Engineering GmbH
„NX Systembetreuer NX 10 / NX 11“
- NXOpen Programmers Guide
- NXOpen .NET API Reference
- Seit NX 11.0: Getting Started with NXOpen (PDF)
- Lokale Beispielprogramme in der NX-Installation
(%UGII_BASE_DIR%\UGOPEN\SampleNXOpenApplications)
- Knowledge Database von Siemens (Stichwort „GTAC“)
- Inoffizielle Internetforen, z.B. <http://ug.cad.de>



Zusammenfassung

- Gute NX-Zusatz-Programme bringen Zeitersparnis und Sicherheit in Prozessen!
- Gute NX-Tools kosten Zeit und Geld, machen sich aber sehr rasch bezahlt!
- Nur Journaling ist ohne Autoren-Lizenz nutzbar! (Einstiegsmöglichkeit...)
- SNAP ist sehr einfach erlernbar!
- Für SNAP ist eine Lizenz notwendig!
- Funktionsumfang von SNAP ist noch eingeschränkt!
- NXOpen, UF-Bibliotheken in .NET sehr umfangreich
- SNAP, NXOpen, UF sind in einem Programm gemeinsam nutzbar
- Für NXOpen, UF in Kombination mit SNAP ist eine Lizenz notwendig!
- Typische NX-Dialoge erfordern eine Lizenz für den „Block UI Styler“

HBB-Engineering GmbH
Salzstraße 9
D-83454 Anger

Telefon: 08656-98488-0
Telefax: 08656-98488-88

info@HBB-Engineering.de
www.HBB-Engineering.de

Vielen Dank für Ihr Interesse!

Haben Sie noch Fragen?

Besuchen Sie uns auf Stand Nr. 32