

최 첨단의 **Embedded software**를 소개합니다.

**NexWave**사의 컴포넌트 기술은 컴포넌트화한 프로그램의 재사용을 통하여 **Embedded device software**개발 효율 증대에 목표를 두고 있습니다.

NexWave 사는 Embedded Software 의 컴포넌트화를 가능하게 하는 툴과 환경 및 제반 기술인 **NSI** (NexWave Software Infrastructure)를 개발 하였습니다. **NSI** 환경에서 만들어진 컴포넌트는 여러 응용 장치에서 하드웨어 플랫폼(Hardware Platform)이나 운영체제(Operating system)에 상관 없이 쉽게 여기 저기에서 다시 사용 할 수 있습니다. 이러한 **NSI** 컴포넌트의 사용은 고객님의 개발 기한을 혁신적으로 단축시켜주며, 개발 비용을 현격하게 절감하도록 도와 드립니다.

**NSI** 는 고객님의 이미 개발 완료하여 보유하고 있는 막대한 프로그램 자산이나 개발 중에 있는 프로그램 일부를 이용하여 컴포넌트화 하고, 그 컴포넌트들을 이용하여 새로이 구성 하여 새롭게 어플리케이션 프로그램을 만들 수 있도록 도와주는 프로그램이며, 툴이며, 기술이며, 통합적인 환경입니다. 또한, **NSI** 는 Embedded device software 에서 컴포넌트 기술을 적용하는데 있어서 하드웨어 성능에 영향을 미치지 않고 Processing 이나 Memory 의 증가도 요구하지 않는 유일하고 적절한 컴포넌트 모델입니다.

**NexWave**사는 1998 년에 가전 제품 개발 시장에서 Embedded Software의 미래의 변화에 대한 전망에 대하여 같은 비전을 공유해온 소규모의 Software엔지니어들에 의해 창립되었습니다. 그리고 그들이 함께 Embedded device software가 당면한 방대한 도전 과제에 대하여 진취적으로 대응하여 야심에 찬 기술 개발을 논의하였습니다.

**NexWave** 사의 성공은 component based software 기술에 뿐만이 아니라, 기업의 가치관에도 있다는 것을 알고 있습니다. 저희 회사는 고객의 입장을 최우선으로 하며, 고객이 개발 일정 단축과 개발 비용 절감의 목표를 달성할 수 있도록 최선을 다하여 고객을 돕고 있습니다.

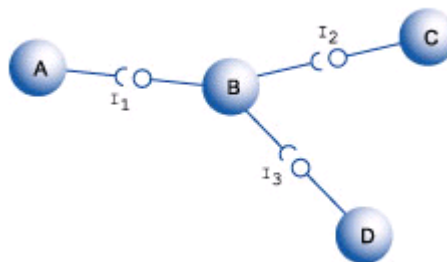
**NexWave** 본사는 지중해에서 10 분 정도 떨어진 남부 프랑스 지방 Montpellier에 위치하고 있습니다. 근래 캘리포니아 실리콘밸리에 새로운 지사를 개설하였습니다. 다른 지사와 아시아의 대리점이 2005 년 중 설립이 예정되어 있습니다.

## Components: 기능 시스템을 이루는데 잘 정의된 인터페이스들을 통해 상호작용 하는 Binary 의 속성

NSI 는 가전 제품 생산 업자들과 함께 수년 이상 협업을 통해 개발 된 가볍고, 적응성 있고, 플랫폼과 운영 체제에 독립적인 component model 입니다. NSI 는 기본 개념들과 법칙들을 최소로 제한함으로써, 빠른 속도로 프로그래머들이 익숙해지는데 도움이 되도록 설계되어 있습니다.

하단의 그림은 NSI component model 의 구성 요소들을 보여주고 있습니다.

- A, B, C, D 는 컴포넌트
- I1, I2, I3 는 인터페이스
- B 는 I1인터페이스에 정보를 제공하고 I2, I3 인터페이스로부터 정보를 받아들입니다. B 의 의존 상태는 C, D 의 export matching 인터페이스들과의 import 결합에 의하여 해결됩니다.



### Components and Interfaces

NSI 는 다음의 속성들로 정리 할 수 있습니다.

- 인터페이스 정의를 내포하고 있는 Binary 속성의 컴포넌트
- Build-time 과 run-time 에서 의존관계 해결
- 컴포넌트간의 적응성 있는 연결
- 컴포넌트간의 직접 통신 (direct function calls whenever possible)
- 리소스와 근본적인 OS 관련한 적은 필요조건

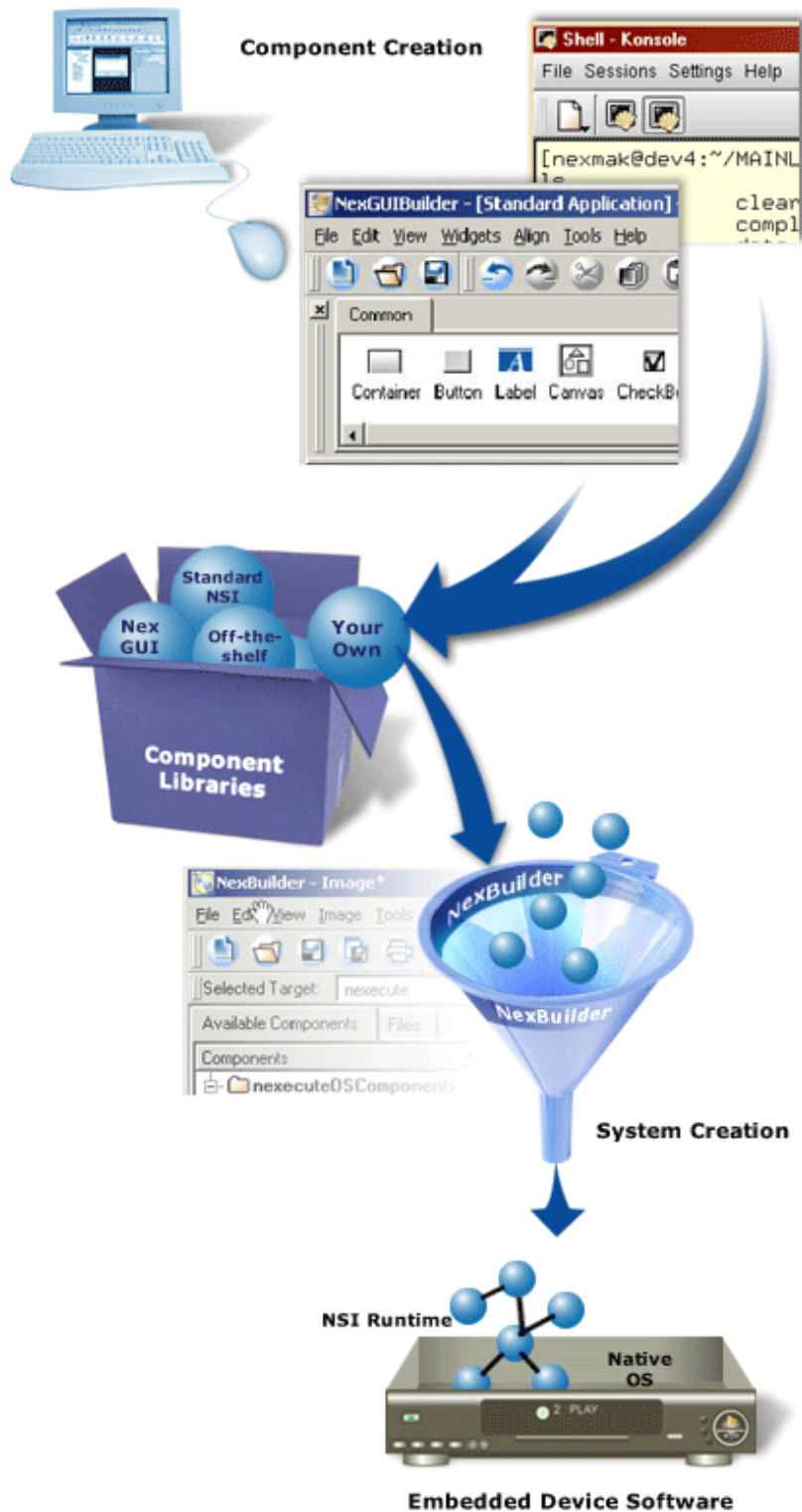
## NSI 의 이점

NSI (NexWave Software Infrastructure) 는 embedded software 산업에 전격적으로 적용시킨 최초의 컴포넌트 아키텍처(구조)입니다. NSI 의 무엇보다 먼저 서로 다른 상품 군 사이를 가로질러 코드의 재사용을 가능하게 하는데 있습니다. 예를 들어, DVD 플레이어 를 위해 쓰여진 컴포넌트가 쉽고 흔적도 없이 위성수신장치에서 재사용 할 수 있는 것입니다. 모든 곳에 재사용이 가능한 컴포넌트들이 만들어지고 추가 되어감에 따라, 더 많은 컴포넌트들의 재사용이 다른 프로젝트들을 가로 질러서 까지 촉진 되어감에 따라, 그러면 개발자들은 되돌아 오는 이익이 증가하는 것을 목격합니다.

|          |   |
|----------|---|
| 이득       | 개발비의 절감                                       |
|          | 발 빠른 시장대응                                     |
|          | 품질 향상   |
| 주목할만한 특징 | 기존 프로그램 자산의 재사용                               |
|          | 운영체제/하드웨어와 독립적                                |
|          | 적은 메모리 추가와 no CPU overhead                    |
|          | 이종의 플랫폼에서의 컴포넌트간의 원활한 통신                      |
| 기타 강점    | 소프트웨어 개발 과정에서 누락된 기능의 보정                      |
|          | Dynamic online update on a live system 리부팅 없이 |
|          | 소프트웨어 제품 납품 과정의 개선                            |

## Products(상품소개)

### Overview and Benefits(간략한 소개와 장점들)



NSI 는 NexWave 의 총체적 핵심 기술 환경으로, 여러 제품 군으로 구성되어 있습니다.

- NexDev 는 NSI 컴포넌트들을 만들어 내는데 필요로 하는 일습의 프로그램 툴로서, 여기서 컴포넌트는 스크래치로부터 쓰여질 수 있고, 기존의 완성 코드로부터 이식 시키든가, NexGUIBuilder 로 만들어 내거나 그 컴포넌트들의 조합으로 만들어 집니다.
- NexBuilder 는 NexDev 을 이용해 이미 만들어진 NSI 컴포넌트들로부터 시스템을 생성 하는 데에 쓰는 그림도구 툴(graphical tool)입니다. 개발자들은 시스템에 원하는 컴포넌트들을 간단히 끌어다 놓기만(drag and drop) 하면 되고, 툴은 필요로 하는 다른 컴포넌트들을 자동적으로 추가합니다.
- NexGUI 는 NSI components 의 모임인데 CE 와 임베디드 어플리케이션에서 그래픽 사용자 인터페이스 프레임워크(a graphical user interface framework)를 제공합니다. NexGUI 는 크기가 작고 빠르며 확장 성이 좋습니다.
- NexGUIBuilder 는 그림 도구 툴(graphical tool)로서 HMI (human-machine interface)를 지향하는 쌍방 소통의 대화식으로 구성되어 있습니다. 타겟이 요구하는 옵션 선택에 따라 상응하여 적합 하게 작동하고, 옵션을 수정해 줌으로써 내부적 작동이 NSI 환경이 맞추어지도록 확장되어 있습니다.

## NSI Runtime

NSI Runtime 는 주어진 운영체제 위에서 NSI 컴포넌트들이 동작하는데 쓰이는 실행파일입니다. 운영체제 와 구조의 조합에 따라 각각 지원하는 NSI Runtime 이 규정됩니다.

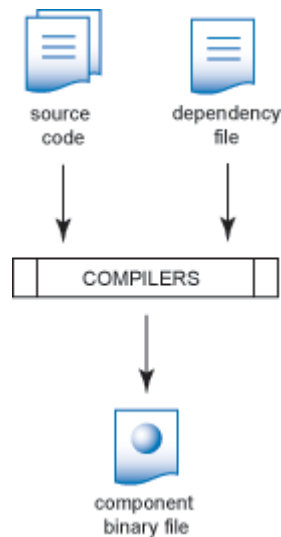
운영체제에서 실행 될 때 NSI Runtime 은

- 연관 관계에 따라 컴포넌트들 활성화시킨다.
- 컴포넌트들과 운영체제간의 연결을 제공한다.
- 어플리케이션의 역할에 따라 NSI component 들을 실행시킨다.

끝으로, NSI Runtime 은 컴포넌트들의 실행에 절대 지장을 주지 않습니다.

## NexDev

NexDev 은 컴포넌트의 인터페이스에 대한 상세 서술 사항(NexWave's Dep 서술방법을 따른), 그리고 컴포넌트 안에 넣을 소스코드를 받아 들입니다. 내용을 자체 내포한 캡슐화된 바이너리 컴포넌트(binary component)는 이것들(인터페이스 서술과 소스 코드)을 결합하여 만듭니다.



NexDev 는 쉽게 컴포넌트의 생성이 가능하도록 미리 정해진 여러 생성 룰들을 따라 수행 되는 인터페이스 컴파일러로 이루어져 있습니다. . NexDev 는 사용자가 이미 가지고 있는 개발 툴들과 개발 환경에 적합 하도록 디자인 되었습니다.

## NexBuilder

NexBuilder 는 사용하기 쉬운 그래픽 툴로서, 개발자들이 간단히 컴포넌트를 드래그 앤 드롭하여 완전한 시스템을 만들어내 낼 수 있도록 해줍니다. 이 툴은 선택해 더해지는 컴포넌트를 자동으로 스캔 하여, 그 컴포넌트의 의존 관계를 만족 시키는데 필요한 다른 컴포넌트들을 끌어옵니다.

NexBuilder 는 가전과 embedded 산업을 염두에 두고 고안되었습니다. 그렇기 때문에 NexBuilder 는 부트가 가능한, 혹은 XIP(eXecute in Place)가능한 이미지를 만드는 유틸리티를 포함하고 있으며, 생성된 소프트웨어 이미지의 메모리 맵을 미리 지정할 수 있는 메커니즘을 포함하고 있습니다.

## NexGUI

NexGUI 는 가전과 embedded applications 에 목표를 두고 graphical user interface framework 을 제공합니다. NexGui 는 NSI 컴포넌트들의 모임으로 구성되어 있으며, frame buffer 를 access 하는

소프트웨어에서부터 데스크톱 레벨 어플리케이션에 이르는 범위까지 실측 가능한 **graphical software architectures** 구축하도록 하여줍니다.

NexGUI 는 최소의 용적과 뛰어난 성능을 제공하면서 유래 없는 호환성을 제공하는 매우 간편한 GUI 를 제공하여 NSI 의 활용력을 증대시킵니다. NexGUI 는 몇 개의 핵심 컴포넌트들로 이루어져 있으며 (image decoders, font decoders, clipper/drawer, widget engine 등) 프로그래머들이 추가하거나 확장시킬 수 있는 작은 부품 형태의 컴포넌트 세트입니다. NexGUI 의 컴포넌트 기반 설계는 개발자들이 자신의 image decoder 나 font decoder 혹은 작은 부품 형태의 컴포넌트들을 간단히 만드는 것으로 새로운 기능을 추가하는 일이 별일이 아닌 일이 되게 만듭니다.

NSI Runtime 과 NexGUI 컴포넌트들은 당신의 개발 컴퓨터에서 사용할 수 있기 때문에, 거의 대부분의 어플리케이션 프로그램이 타겟 하드웨어를 기다리지 않고 컴퓨터 상에서 개발할 수 있습니다.

### NexGUIBuilder

NexGUIBuilder 는 NexGUI framework 을 사용해서 쉽게 컴포넌트들을 만들어 내도록 하여주는 graphical tool 입니다. 사용자의 요구에 맞출 수 있는 NSI 환경을 내부적으로 동작하게 하는 NexGUIBuilder 의 독보적인 능력은 고맙게도, 실제로 WYSIWYG (보이는 대로 얻는 what you see is what you get) 행태를 제공하고 그리고 고객의 요구에 부합 시킬 수 있게 합니다: 똑 같이 반영된 것을 타겟에서 얻을 수 있도록 주어지는 NSI 환경의 디폴트를 수정할 수 있습니다.

NexGUIBuilder 는 고객 자신의 소프트웨어 구조 속에 꼭 맞도록 요구하는 종류의 컴포넌트를 만들어내기 위하여 사용자의 요구에 맞출 수도 있습니다.

### LibTRON

LibTRON 은  $\mu$ ITRON 3.0 나 4.0 상에서 쓰여진 기존 코드를 모든 NSI 환경(e.g., Linux, Windows, etc.)에서도 동작할 수 있도록 하여주는 제품입니다. LibTRON 은 configuration tool 과 NSI 컴포넌트로 이루어져 있습니다. LibTRON 컴포넌트는 어떠한 NSI 환경에서도 이용이 가능한 RTOS abstraction 위에 설치되어 있습니다. LibTRON 컴포넌트는 NSI 의 개념을 보여주는 좋은 실 예입니다: NSI 가 realtime services 를 제공하고 동시에 OS 에 독립적 이도록 컴포넌트를 작성 할 수 있는 것을 보여줍니다.

## Market Focus

CE (consumer electronics) 시장이 특히 NSI의 이점에 대하여 필요의 압력을 받고 있습니다. 이러한 필요의 압력은 가전 생산자(OEMs), 독립된 SW 납품업자(ISVs)와 반도체 공급자 등 모두 마찬가지입니다.



여기에 CE (consumer electronics) 시장이 특별히 NSI의 이점에 대하여 필요의 압력을 받고 있는 네 가지 주된 이유가 있습니다.

### 가정의 디지털화

가정의 디지털화 출현은 CE 생산자들에게 소프트웨어에 대한 큰 부담을 안겨주고 있습니다. 그리고 디지털 미디어로의 이동 또한 마찬가지로 큰 부담입니다. (예: DVD, MP3 players, digital television, digital cameras, camcorders 등.). 소비자들은 까다롭게 점점 더 진보된 기능들을 요구합니다. 소비자의 요구를 충족시키기 위해 하드웨어 보다는 차라리 소프트웨어로 기능을 추가시키는 비중을 늘리는 쪽으로 CE 판매자들은 “디지털화”를 선택 하였습니다. 그 결과 요즘의 CE 사업체의 역할은 하드웨어에서 소프트웨어의 제공으로 이전 되어 가고 있습니다. 업계에서 이렇게 극적으로 변해가는 추세를 따르는 것이 오늘날 CE 회사들이 당연한 주된 과제 중 하나입니다. NexWave의 컴포넌트 기반 기술은 이러한 CE 사업의 난제들을 극복하기 위해서 디자인되었습니다. 특히 여기서 연관하여 언급할 것은 부서 간의, 심지어는 회사간에서의 “코드 재사용”의 열린 기회(높은 가능성)입니다.

### 장치들의 통합

CE 산업에서 주목되는 장치간의 통합이 논거 되고 있습니다. 이제는 내장된 디지털 카메라가 없는 모바일 폰이나, 휴대 저장 장치가 없는 디지털 카메라를 구매하기 힘들어졌습니다. 근래 CE 상품들은 서로 아주 작은 차이밖에 없이 비슷하게 연계되어 있습니다.

좀 더 자세히 말하자면, 가전 시장은 큰 제품 군으로 구성되어 있는 다차원적 점진 발전체인 데, 여기서 소프트웨어가 제품들을 점진적으로 이끌어 가는 역할을 합니다. TV를 예로 들어 보면, 작은



아날로그 텔레비전에서 시작하여 보다 진보된 모델을 거쳐서 통합된 디지털 TV 로, 그리고 중국엔 DVD 나 하드 디스크 구동, 그리고 녹화 기능까지 포함 하는 통합된 시스템으로 점진 발전합니다. 물론 약간의 경우에는 점진의 반대 끝 쪽인 퇴조로 소프트웨어가 이끌어 가기도 합니다만, (저 예산 휴대용 TV 와 최첨단의 디지털 플라즈마 TV) 대부분의 경우 소프트웨어가 점진의 연속성을 갖고 조금씩 가까운 쪽으로 발전해 나가는 역할을 하게 요구 됩니다. 점진 발전체인 가전 시장은 복합적이고 다각화된 다차원적 방향으로 진화되어 갑니다: 고려하여 보면 대부분의 DVD 플레이어는 오디오 CD 를 동작시킬 수 있는 기능을 가지고 있고 -- TV/DVD 콤보 제품은 TV, DVD player 와 스테레오 등의 점진 향상된 성능들이 모두 연결된 점진 발전 체의 결과물 입니다. 오늘날, 전체 가전 제품의 영역은 하나의 다차원의 점진 발전의 연계 안에 놓여 있습니다.

10년 전만 하더라도, 각각의 CE 상품들에 필요한 소프트웨어는 소프트웨어 원형으로부터 (아니면, 적어도 각각 같은 상품 군으로부터) 각 상품 군에 대하여 개발하는 것이 실용적 이었습니다. 하지만, 이러한 작업은 점점 더 비현실적이고, 비효율적인 방법이 되어가고 있습니다. 이제는 CE 생산자들이 자신의 제품 영역을 넘어 다른 영역의 제품 군들과 서로 프로그램을 공유하는 방법을 찾아 내야만 합니다.

### 거듭되는 개발의 불필요

CE 사업체들간에 서로의 소프트웨어를 공유 할 수 있게 되는 것은 점점 더 매력적으로 되어가고 있습니다. 다른 생산자들이 각각이 막대한 양의 프로그램을 원형의 스크래치 파일로부터 다시 개발 하고 있습니다. - 각각 다른 생산자가 매번 되풀이 하여 개발하는 소프트웨어의 약 80% 이상이 별반 차이가 없는 것들 입니다.

### 바이너리 컴포넌트의 공유

코드를 가져다 편집 적용하는 것이 컴포넌트를 가져다 적용하는 것보다 훨씬 비효율적임에도 불구하고, 물론, 단순히 소스 코드를 가져다 편집 적용시키는 것이 코드의 재사용의 한 방법일 수는 있습니다. 사실 엄정하게 말하면 컴포넌트 모델은 독립적으로 개발된 코드 조각들을 쉽게 조립하게 해서 프레임웍을 하도록 하는 것입니다. 더욱이, NSI 컴포넌트를 사용 하는 것이나 소스코드를 편집 하는 것이 서로 배타적 개념은 아닙니다.: 하지만 소스 코드 연계한 방법의 컴포넌트의 제공에는 코드 공개에 대한 예방이 전혀 없습니다.

### 개발비의 절감

소프트웨어 엔지니어의 고용 비용은 내려가지 않고, 그리고 많은 연구 보고서들에 의하면 가장 뛰어난 엔지니어 들조차도 하루에 양질의 양산용 소프트웨어 코드로 따지면 단지 10여줄 정도 쓸 수 있는 것을 보여 줍니다.

## 발 빠른 시장대응

대다수의 가전 관계 회사들에게 신속한 시장대응이 개발비용보다 훨씬 더 중요하여, 소프트웨어의 재사용은 그 이유가 되고 동시에 더 낮은 비용의 개발 방법 이기도 하고, 또한 소프트웨어의 재사용은 소프트웨어 전체를 다시 쓰는 것보다 훨씬 빠른 방법 이기도 합니다.

## 품질

고객들이 요구하는 품질 수준을 얻어내야 하는 것이 소프트웨어 프로젝트 지연의 가장 일반적인 이유입니다. (많은 경우의 지연 소프트웨어 프로젝트들의 지연 이유는 장기간에 걸친 bug 수정 과정에 기인합니다.) 다양한 상황에서 검증 된 재사용 가능한 소프트웨어가 **scratch**로부터 얻어진 소프트웨어보다 훨씬 품질이 높을 것입니다.

## 기존 프로그램의 재사용

과거에 만들어진 소프트웨어로부터 이미 가지고 있는 라이브러리, 모듈 또는 소스 파일들은 모두 컴포넌트 기반 인프라스트럭처에서는 가치 있는 자산으로 되어 축적 됩니다. **NSI** 는 기존의 **compiler** 들을 교체하지 안으면서도 그것들과 연계하여 작동 하고, 기존 **object file** 들을 바이너리로 엔코딩된 컴포넌트의 입출력 인터페이스들과 연결(linking)시킵니다. 인수(변수, 인터페이스)의 재배치를 새로운 컴포넌트의 설계에, 그리고 새로운 컴포넌트 인터페이스의 요구에 한정 시킴으로써, 컴포넌트화에 들어가는 비용과 노력의 절감을 지향 하고 있습니다.

## 운영체제/하드웨어 호환성

**NSI** 자체는 어떤 하드웨어에도 구매 받지 않으며, 다양한 운영체제에서 이용 가능합니다. 몇몇의 **NSI** 컴포넌트들은 완벽하게 **RTOS APIs(Application Program Interfaces)** 을 제공합니다. (예로 **memory, mailbox, semaphore, mutex** 등의 컴포넌트들) 이러한 **APIs** 들이 **RTOS** 에 없애이지 않는 컴포넌트를 작성 할 수 있도록 해 줍니다.

더 나아가서, **NSI** 컴포넌트가 다른 **RTOS** 의 **APIs(TRON 같은)**를 시뮬레이션 하여 제공합니다. 이점으로 인해서 **TRON** 이나 다른 **RTOS** 의 기존 **SW** 자산으로부터 만들어진 컴포넌트가 **NSI** 환경에서 작성된 컴포넌트와 함께 전혀 무리가 없고 흠 없이 동작하도록 컴포넌트를 작성하게 하는 강력한 환경이 가능한 것입니다. 때문에, 다양한 종류의 **OS** 들과 하드웨어를 넘나드는 이식성이 높은 컴포넌트를 작성하기가 매우 쉬운 것입니다. 더 나아가서, **NSI** 는 **framebuffer (GUI)**를 위한 비현실적 일듯한 컴포넌트, **keyboard, file access, IP communication (sockets), etc** 즉 드라이버도 제공합니다. 끝으로, **PC(Windows or Linux)**상에서 **CE** 응용프로그램을 개발 하면서 손질하고 에뮬레이트 하는데 단지 몇 개의 칩셋 드라이버만이 필요할 뿐입니다.

## 최소한의 메모리 추가와 no CPU overhead

메모리 추가는 NSI가 발표 하였듯이 runtime 에 만 한정 시켰습니다. (150 KB 이하) 다른 평균적인 소프트웨어 모듈들과 비교 했을 때, 컴포넌트의 메모리 추가는 최소를 유지하면서 컴포넌트 인터페이스 엔코드하는 단 한 곳만 제외하고 없습니다. 또한, CPU의 부하는 실행하는 동안ダイナ믹하게 조절되어서 컴포넌트 모델에서 CPU의 부하는 제로로 됩니다.

## 다른 플랫폼에 있는 컴포넌트들과의 쉬운 통신

NexWave사의 DNSI (Distributed NSI)가 네트워크를 통하여 다른 컴퓨터상에 속해 있는 컴포넌트들과 정보를 주고 받는 통신을 가능하게 하고, 같은 방식으로 DNSI (Distributed NSI)는 같은 머신 안에서 컴포넌트들 간에 정보를 주고 받는 역할로 프로그래머에게 나타납니다. 이것이 "location transparency"로 알려진 방식입니다. 같은 컴포넌트들은 같은 머신에서 나란히 놓이거나 혹은 다른 머신에 걸쳐 놓거나 하여 재 컴파일 없이 그대로 사용될 수 있습니다. 실제의 네트워크 시스템들처럼 네트워킹이 잘 되므로 해서, 어떤 내부 연결과 관련하여((i.e., asymmetrical or NUMA multiprocessor systems) 불균질한 이종의 내부 처리기를 포함하고 있는 가전 장치들에 유용합니다.

## 개발 과정에서 누락된 기능의 수정

NSI는 개발 완료된(출시된) 시스템에 새로운 컴포넌트를 load시킬 수 있습니다. (컴포넌트를 로드할 수 있는 어떤 매체가 있어 제공되면 - 다시 말하면 DVD를 이용하거나 인터넷을 통해 다운로드를 하거나, 시스템 개개의 연결 방법으로) 그런 매체가 실장되어 있다면 이미 개발 완료된(출시된) 시스템에 새로운 서비스를 제공할 수 있고, 서비스 제공자, 개발자와 최종 소비자 간에 실시간 결재망 전 과정을 설치할 수 있습니다. 같은 개념으로 설계 개발 과정에서 늦어져서 누락되었던 것이나 기능의 추가로 기인한 컴포넌트를 추가 시키는데 아주 실용적이고 유용합니다.

## Dynamic on-line update

NSI는 리 부팅 없이, 살아 동작하고 있는 시스템상에서 바로 수행 될 컴포넌트들의 교체 실행까지도 허용합니다. 이러한 점은 바로 고객의 불편 없이 소프트웨어를 간단히 업데이트하는데, 멈출 수 없을 정도로 매우 중요하고 절대가치의 업무를 수행하고 있는 시스템의 버그를 수정할 때 유용하게 사용할 수 있습니다.

## 소프트웨어 제품의 납품 과정의 개선

컴포넌트로 구성된 소프트웨어의 공급과 납품이 훨씬 더 소스파일들의 임의적인 모음이나 블록들을 크게 한 덩어리로 하여 납품 하는 것 보다 더 실리적이고 유용합니다. NSI는 완벽하게 소프트웨어를 자체 내장시킨 컴포넌트를 가능하게 하고, 단일 파일로서 라이브러리나 소프트웨어 시스템의 일부로

납품을 허용합니다. 개발자가 시스템을 구성할 때 하나 하나의 헤더파일이나 컴파일 환경에 일일이 액세스 하여 고려 해야 하는 일이 없도록 문제점을 줄여줍니다.