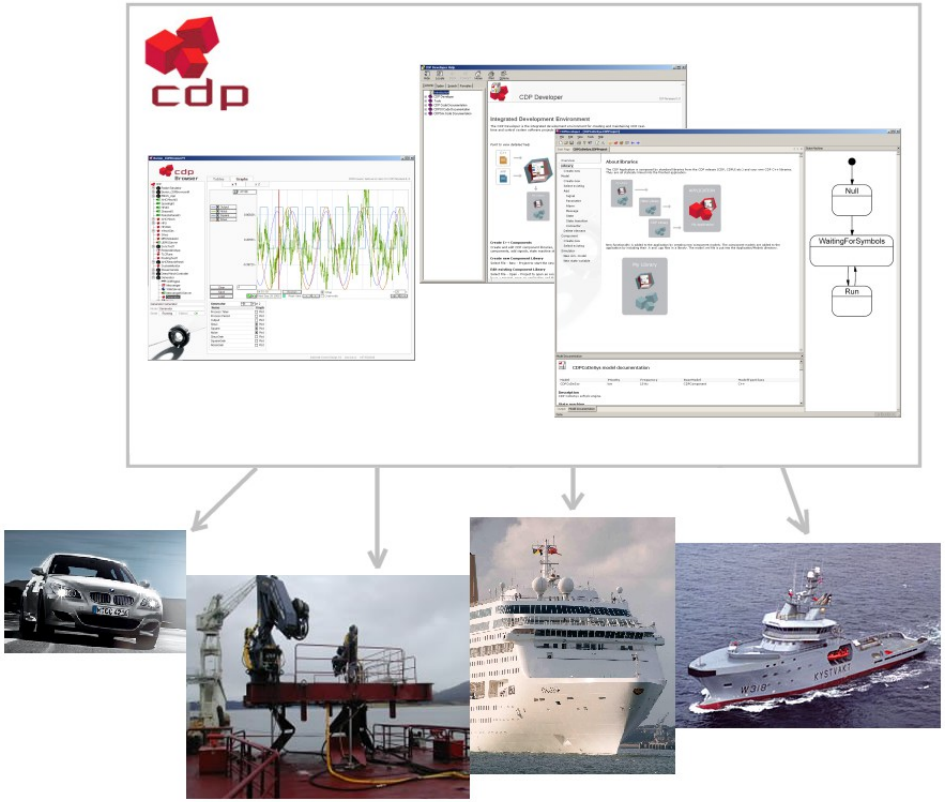
cdp

control development software



control development software

CDP dramatically changes the way control systems are built:

- Significantly reduced development time
- New possibilities giving increased performance and functionality of controlled equipment

control development software

- Designed for advanced and complex control systems, demanding the best possible performance, flexibility and safety

control development software

- Flexibility of C++ and efficiency of high- level application development
- Efficient software tools assist the development process from C++ code generation to testing and maintenance
- Middleware layer provides generic functionality and services, such as communication, allowing you to focus entirely on application development

control development
software

CDP is:

- Component based
- Distributed
- Platform independent

CDP foundation

We wanted to solve typical problems occuring in real- time and embedded development projects:

- Time consuming
- Special knowledge required
- Risc; will it be completed, is it stable now..
- No reuse of methodology and software
- Awkward and less known tools
- Documentation out of date
- A considerable amount of available resources used on application independent infrastructure

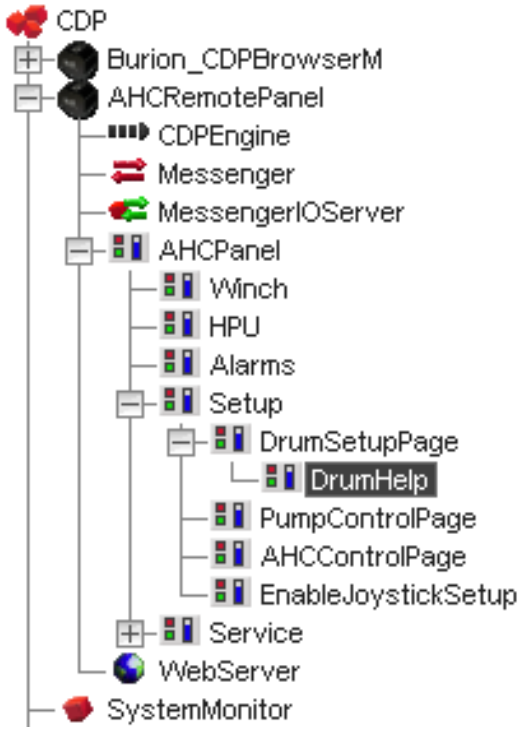
CDP foundation

CDP is based on ideas found to increase efficiency of development and make life easier for the developer:

- Freedom to make my own solutions.
- Infrastructure: If all effort can be focused on application development, the end result will be much better.
- Develop and test on workstation.
- A simple, defined method for C++ implementation.
- Simulate parts of the physical process.
- Efficient tools for signal analysis and process state variable monitoring.

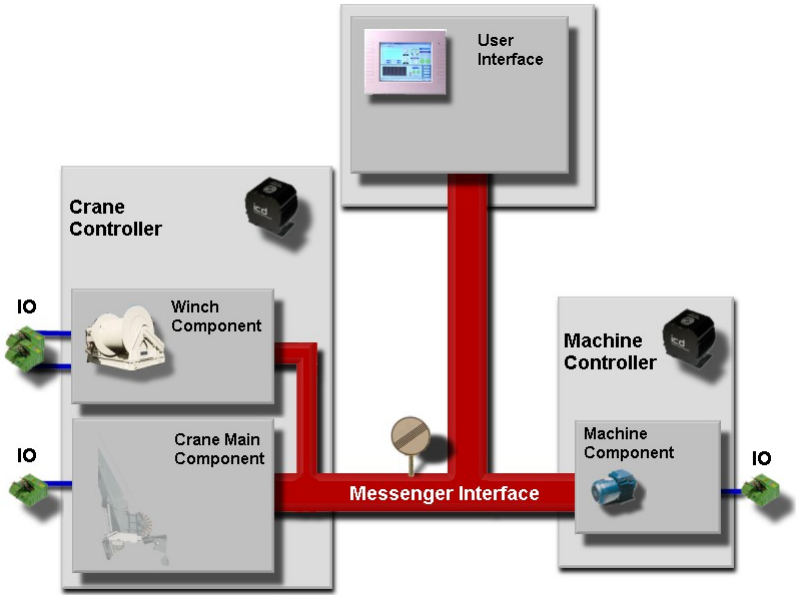
Component Architecture

- Components in a hierarchical structure
- Implementation reduced to component creation
- Components are actually reusable



Distributed

- Logical structure independent of physical
- CDP provides real- time communication between components

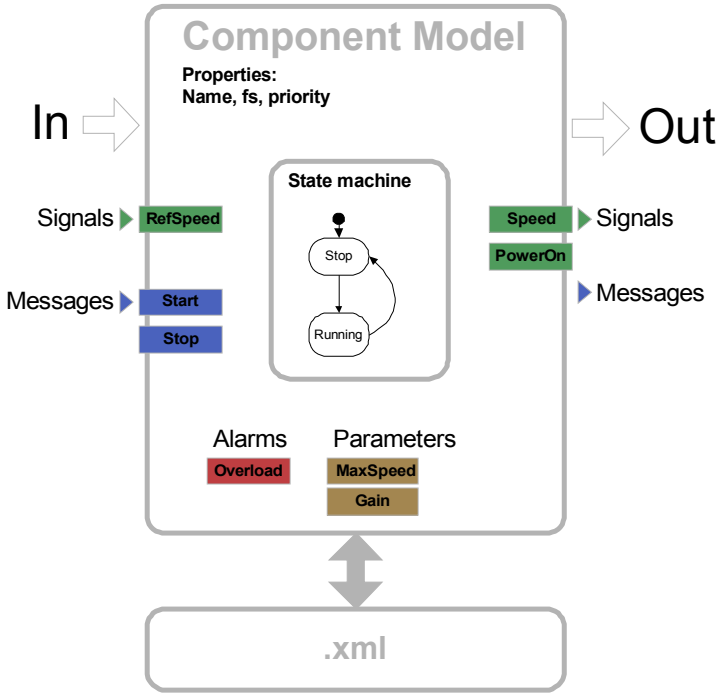


Distributed Development

- The distributed, component- based architecture makes it easy to distribute development to separate teams or individuals
- Typically, a new component will have a clear and simple specification of its interface, since all lower- level interfacing and communication issues are solved by CDP

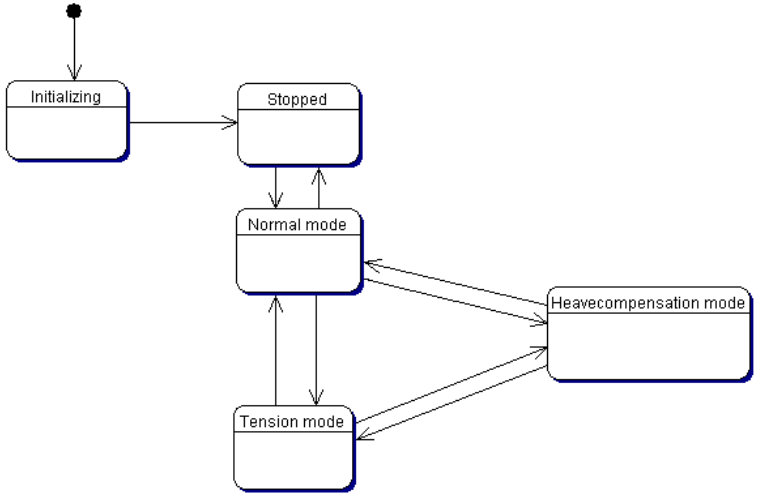
Component model

- State machine and periodic process
- Signals
- Messages
- Alarms
- Parameters
- Persistent in .xml
- Consistent code structure



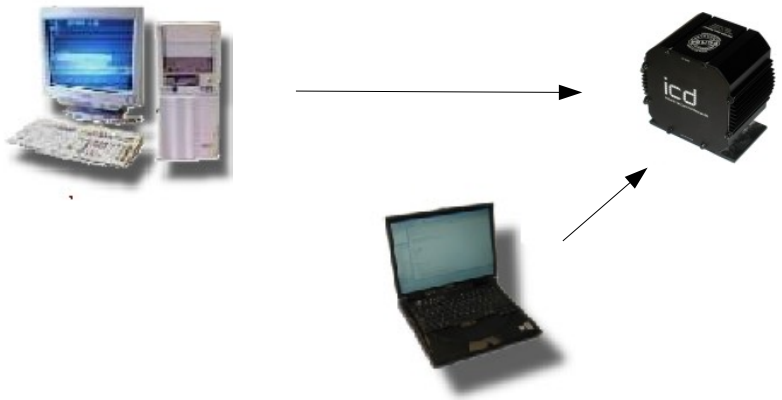
Component model

- State machine with State Transitions
- Can also create new threads



Platform independent

- Supports Windows and real- time operating system
- Can run application tests on the workstation
- Same application code on workstation and target



Use exisiting code

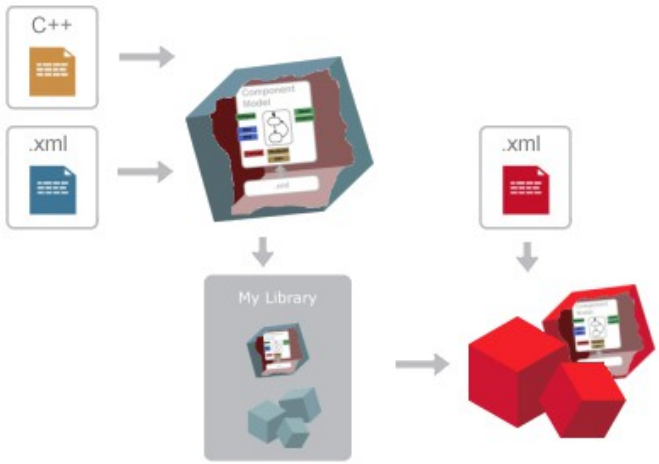
- Existing code can be integrated in to components
- Adds CDP functionality to old programs, bringing them into the future

Signals and parameters

- Signal member objects let you monitor and transfer values used in the C++ code in real-time
- In the code, both Signals and persistent parameter objects are used as if they were primitive data types like int or double

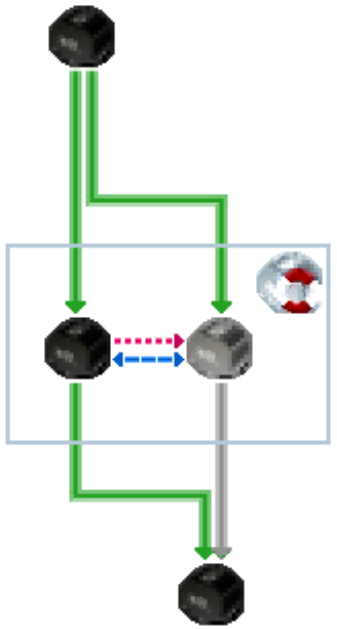
Development process

- C++ programming of new components result in model libraries, which are selected and linked to executable binaries
- No need for C++ programming when distributing and configuring applications which consist of components already developed and tested
- CDPDeveloper generates all necessary framework code for new CDP components
- Applications are created by instantiating and configuring components from model libraries



Option: CDP Redundancy

- Full redundancy functionality and the power of C++ programming
- Can run on different hardware and operating system
- Performance only limited by hardware
- Flexible set-up to suit any application requirements



Option: CDPUI

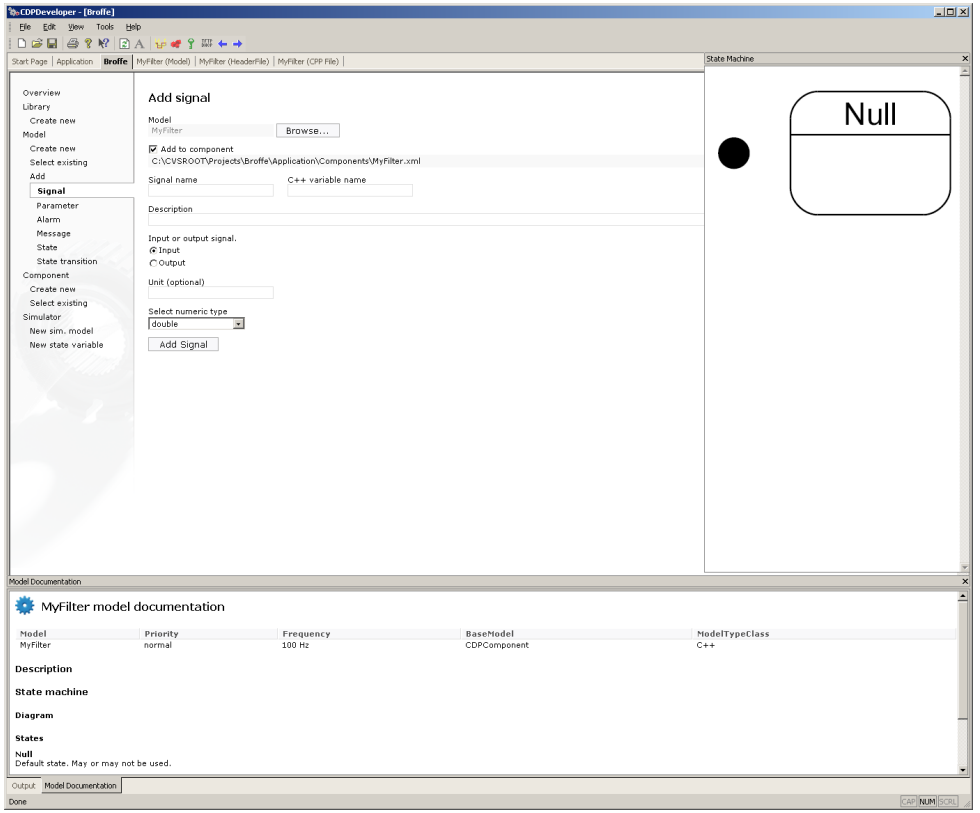
- Graphical user interface for control systems
- Integrated with control software
- Makes traditional ‘HMI’ obsolete

Option: CDPSim

- Dynamic simulation toolkit
- Simulate non-linear and complex models
- Integrated with control software
- Hardware-in-the-loop
- Reduce risc by simulating parts of the physical process

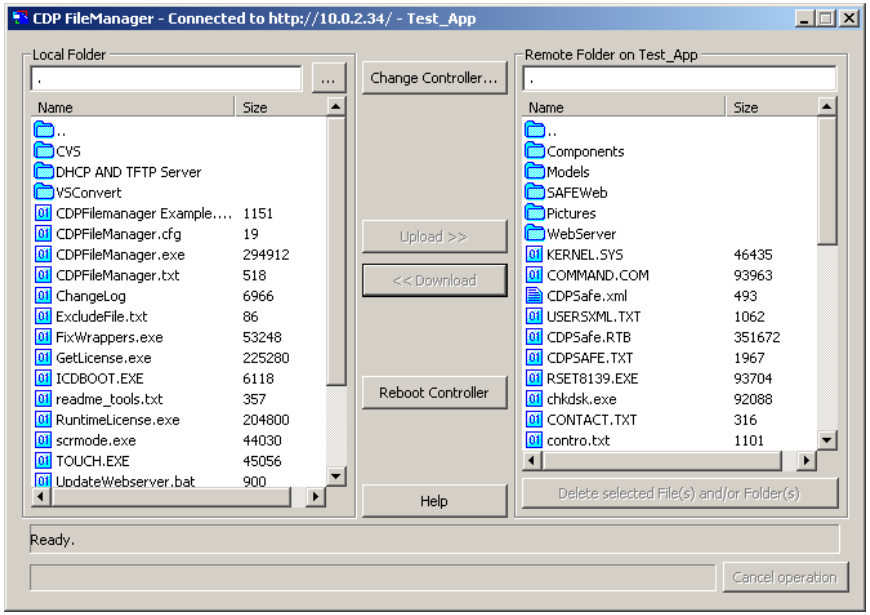
Tools: CDPDeveloper

- Development environment for CDP
- Code generator
- View and edit .xml and C++ files
- View and generate documentation



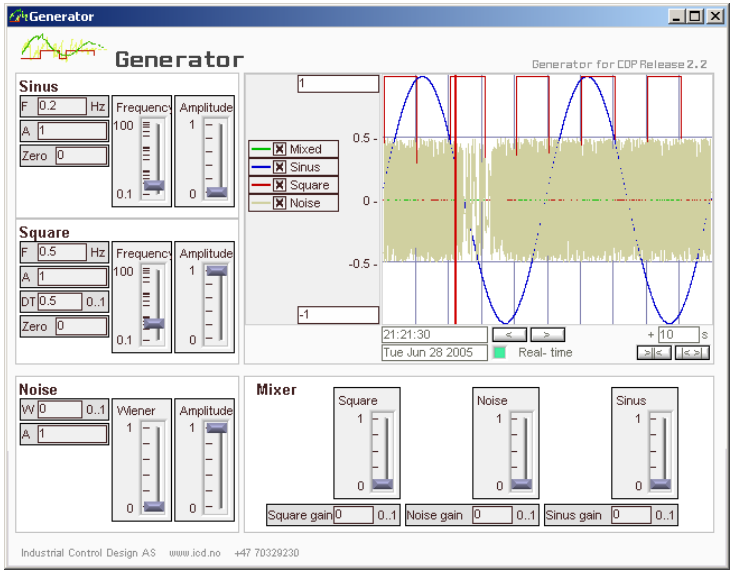
Tools: CDPFileManager

- Upload and download files to target controller



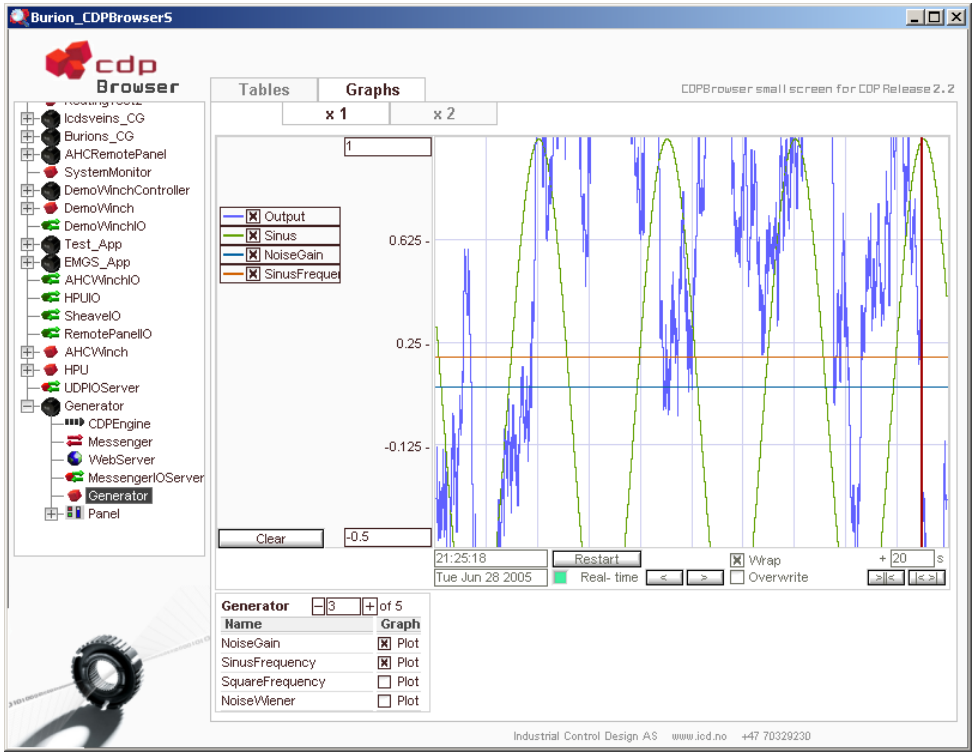
Tools: Generator

- Signal source for testing and simulation
- Easily extendable to suit your custom needs



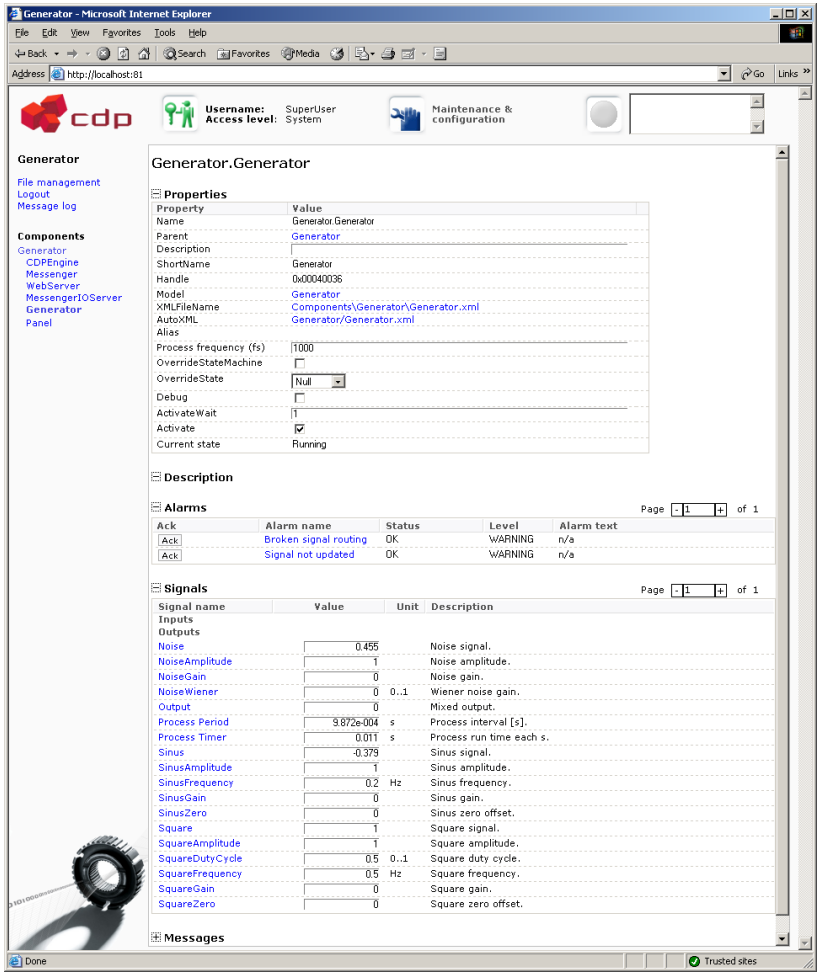
Tools: CDPBrowser

- Browse all CDP applications and components
- Monitor and edit signals, parameters, alarms
- Send messages



Tools: WebServer

- Monitor and edit signals, parameters, alarms
- Edit properties
- View and generate documentation



Case: AHC Platform

- 3 degrees of freedom active motion- compensated platform
- Non-linear, multivariable control
- Extensive use of simulation

Case: AHC Platform

- Graphical user interface and control on same computer
- Distributed IO
- CDP makes it possible to use standard off-the-shelf hardware for this high-performance application

Case: Software development

- Using simulator to test control application
- Everything runs on workstations

Case: Testing

- At first, all mechanical parts are simulated
- Gradually enabling more physical parts
- Using simulated vessel movement to test on-shore

Case: AHC Platform in-use

Industrial Control Design AS

www.icd.no